

FY 2012 US DOE SC
Advanced Scientific Computing Research
Joule Software Metric: Q2 Update

August 13, 2012

APPLICATIONS

Drekar

Roger P. Pawlowski (Oak Ridge), John N. Shadid (Sandia), Paul T. Lin, Eric C. Cyr, Tom M. Smith

VASP - Materials Project

Paul Kent (Oak Ridge), David Skinner (NERSC), Gerd Ceder (MIT), Kristin Persson (California)

NIMROD

Carl Sovinec (Wisconsin), Charlson Kim (Washington), Xiaoye Li (California), Ping Zhu (Wisconsin)

QMCPack

Jeongnim Kim (Oak Ridge), Paul Kent (Oak Ridge)

Active Participants -TBD

Cihan Ackay (Washington), NIMROD problem three

ORNL NCCS and NERSC staff.

Jack Wells, NCCS - Oak Ridge

Contacts

Daniel Hitchcock, US DOE ASCR

Lucy Nowell, US DOE ASCR¹

Kenneth J. Roche, PNNL, U. Washington^{2 3}

¹Lucy.Nowell@science.doe.gov

²High Performance Computing, Computational Sciences and Mathematics Division, Pacific Northwest National Laboratory, kenneth.roche@pnl.gov

³Nuclear Theory Group, Department of Physics, University of Washington, k8r@uw.edu

Contents

Q2 Overview	9
Project Objectives, Methods	15
Computational Science Capabilities, Q2 Metric Status	22
0.1 Drekar: Next Generation CFD Simulations of Turbulent Fluid Flow and Heat Transfer in PWRs	23
0.1.1 Introduction	23
0.1.2 Background and Motivation	23
0.1.3 Capability Overview	24
0.1.4 Science Driver for Metric Problem	25
0.1.5 Software Description: Model and Algorithm	27
0.1.6 Q2 Baseline Problem Results	37
0.1.7 Developments, Enhancements, and Q4 Benchmark Results	49

0.1.8	Q4 Baseline Problem Results	49
0.1.9	Application Summary of Results and Analysis	49
0.2	VASP - Materials Project	50
0.2.1	Introduction	50
0.2.2	Background and Motivation	50
0.2.3	Capability Overview	50
0.2.4	Science Driver for Metric Problem	51
0.2.5	Software Description: Model and Algorithm	51
0.2.6	Q2 Baseline Problem Results	52
0.3	NIMROD	54
0.3.1	Introduction	54
0.3.2	Background and Motivation	55
0.3.3	Science Driver for the Metric Problem	55
0.3.4	Software Description: Model and Algorithm	56
0.3.5	Q2 Baseline Problem Results	58
0.3.6	Science Driver for Q2 Baseline Problem Two	60
0.3.7	Q2 Baseline Result -Problem Two	62
0.3.8	Science Driver for Q2 Baseline Problem Three	62

0.3.9	Q2 Baseline Result -Problem Three	63
0.4	QMCPACK	69
0.4.1	Introduction	69
0.4.2	Background and Motivation	69
0.4.3	Science Driver for Metric Problem	70
0.4.4	Q2 Baseline Problem Results	70
	Bibliography	73
	Appendices	73
0.5	A. Software Libraries	74
0.5.1	Modules Available on the Target Architectures	74
0.6	Machine Event Data Collection	81
0.6.1	KRP Machine Event Data Collection API	81
0.6.2	KRP Use Examples - Level 3 BLAS	82
0.6.3	Automatic instrumentation in GNU environment	114
0.7	Application Inputs, Scripts, Environments	126
0.7.1	Drekar	126
0.7.2	Materials Project - VASP	145

0.7.3	NIMROD	167
0.7.4	QMCPACK	192

List of Figures

1	Weak and strong scaling of LAMMPS in hybrid CPU-GPU environment for MD. . .	19
2	eSTOMP: Lustre-based I/O parameter study.	20
3	Problem geometry for mixing vane and center tube of 600K CUBIT HEX element.	26
4	Swirling Flow by mixing vane and idealized.	27
5	The cut-away view of the HIT-SI experiment.	68
6	Detail of CPU time (μ sec) in three core stages of NIMROD time-advance.	68
7	Speedup of Titandev (XK6 with GPU) over Titan (XK6-CPU) and Monte Rosa (XE6).	71

List of Tables

1	eSTOMP: Lustre I/O parameter study -best parameters.	20
2	Navier Stokes Equations.	28
3	Spalart-Allmaras RANS Turbulence Model Equation.	32
4	Intrinsic-time-scale stability parameters for transport equations.	34
5	PAPI counter results for Swirling Jet simulation.	38
6	Time monitor results for swirling jet simulation for .1 seconds.	38
7	VASP baseline performance results for Hopper at NERSC.	52
8	VASP baseline performance results for Jaguar at OLCF.	53
9	Hardware-counter for the 24-core medium linear computation.	59
10	Hardware-counter for the 128-core large linear computation.	59
11	Hardware-counter for first segment of the nonlinear computation.	60
12	Hardware-counter for last segment of the nonlinear computation.	60
13	MC samples generated per second on Jaguar, TitanDev, and Monte Rosa.	71

14	Theoretical complexity of $C(m,n) \leftarrow \alpha A(m,l)B(l,n) + \beta C(m,n)$	88
15	Measured machine events of two sequential work phases in loop.	112
16	Measured machine events of multiple parallel work phases in loop in distinct sub-groups.	112
17	Measured machine events of threaded parallel work phase.	113

List of Algorithms

1	krp_init()	82
2	krp_rpt_init()	83
3	krp_init_sum()	84
4	krp_rpt_init_sum()	85
5	krp_stat()	86
6	krp_rpt()	87
7	<i>Two sequential work phases in Loop: calculate $C \leftarrow \alpha AB + \beta C$</i>	89
8	<i>Multiple subgroups with parallel work phase in Loop: calculate $C \leftarrow \alpha AB + \beta C$</i>	104
9	<i>Multi-threaded work phase: calculate $C \leftarrow \alpha AB + \beta C$</i>	105

Q2 Overview

Scientific Merit The software applications for FY12 are briefly described along with a brief description of the baseline problem and measured machine events, and a statement of our pursuits.

- Drekar::CFD is a next-generation massively parallel multi-physics simulation code that is being developed for computational simulations of turbulent fluid flow and heat transfer in nuclear fission reactor-cores. The Drekar::CFD code is designed with advanced computational fluid dynamics (CFD) capabilities for large-eddy simulation (LES) and Reynolds averaged Navier-Stokes (RANS) turbulence modeling. The solution algorithms in Drekar::CFD are based on implicit, strongly coupled, Newton-Krylov (NK) methods that also enable the development of advanced adjoint-based integrated error-estimation and sensitivity analysis that can be used for uncertainty-quantification methods. The scalability and efficiency of the NK solution method is provided by physics-based and fully-coupled algebraic multilevel preconditioners. The Drekar::CFD code has been run on 10K cores and the solution methods have been demonstrated up to 144K cores. Currently Drekar::CFD is being applied to study the complex turbulent fluid flow and heat transfer characteristics of fuel rod bundles, and also to provide information about the highly unsteady turbulent pressure forces acting on fuel rods that causes rod vibration. Fuel rod vibration has been shown to deteriorate the rod cladding material that mechanically confines and isolates the nuclear fuel rod material from the cooling flow, and therefore the exterior reactor-containment environment. High fidelity turbulence modeling will lead to a better understanding of rod excitation phenomena and has the potential to improve clad time-to-failure, enhance heat transfer, and improve overall reactor core performance. The Drekar::CFD code is being developed in collaboration with the Consortium for Advanced Simulation of Light Water Reactors (CASL), which is a new U.S. Department of Energy Innovation Hub that is investing in the development of a virtual nuclear reactor toolkit. Our efforts will focus on improved scalability of the nonlinear solvers on the Cray XK6 which essentially implies that we will improve the performance of the Krylov library used from Trilinos for a 3-by-3 rod assembly with realistic input conditions.
 - The Q2 baseline problem is a swirling jet designed to mimic the fluid flow in the pin assemblies of a nuclear reactor just downstream of the mixing vane and central tube, however is not conflicted by classified or proprietary information. The problem includes 1.3 billion unknowns, was executed on 131K PEs of the Cray XK6 at the OLCF retiring 6349855288448493501 total instructions and 130486495829888641 floating point operations in 21,381.85 seconds. The goal is to introduce a speedup at fixed hardware scale. More details can be found here 0.1.4, and about software issues in the appendix here 0.7.1.
- The NIMROD code (<http://nimrodteam.org>) is a flexible computational tool for numerical studies of extended magnetohydrodynamics, which includes MHD, two-fluid plasma modeling, minority ion kinetics, and nonlocal parallel kinetics. It is used to investigate macro-

scopic dynamics of magnetized plasmas, where instabilities lead to topological changes in the magnetic field with subsequent effects on confinement. The mathematical foundations are the nonlinear PDE systems for the MHD and two-fluid plasma models. Problems are solved in the time domain and in three spatial dimensions. Multiple temporal and spatial scales make the problems extremely stiff. For MHD computations, the dominant terms lead to self-adjoint differential operators. In the two-fluid model, the differential operators are not self-adjoint. Kinetic modeling can be incorporated as additional closure information for the fluid-moment evolution equations. The minority-ion and parallel kinetics models share an integral nature of accumulating information along characteristic trajectories. The former is solved numerically in phase space (with PIC), and the parallel kinetics is handled semi-analytically. The NIMROD code uses a flexible, spectral finite-element representation of the poloidal plane and finite Fourier series for the periodic toroidal direction. Projection operations for the finite elements are computed with numerical quadrature. Domain decomposition for parallel computation is applied to all three spatial dimensions. To address the stiffness resulting from a wide range of physical time-scales, the algorithm combines implicit and semi-implicit techniques in the numerical time-advance. In practice, this requires solution of matrices with large condition numbers (due to the range of time-scales) at each time-step. In fully nonlinear computations, the matrices are dense with respect to Fourier component but sparse in the poloidal plane. Matrix-free Krylov iteration is used to avoid direct computation of the convolution matrices that couple the Fourier components. The matrix-vector products needed for matrix-free iteration are formed with FFTs for efficiency. Preconditioning for either CG (symmetric systems) or GMRES (non-symmetric systems) is very important and uses splitting based on Fourier-component blocks. Coupling among solution coefficients of the same Fourier index, over all finite elements, is typically inverted with SuperLU_DIST as the block-diagonal step after condensation over element interiors to reduce the size of the algebraic systems. An option to apply a block Gauss-Seidel-like pass with limited coupling among Fourier components is also available. We will study edge localized modes (ELMs) in tokamaks since this problem stresses more than one part of application software and will benefit from both improved scalability, leading to greater resolution, and improved speed to allow computation over more physical time. The problem will be studied with either the MHD or the two-fluid model to stress the features of the solver to greater or lesser extents. We would like to pursue the enhancements on both the Cray XE6 at NERSC and the Cray XK6 at the OLCF.

- The Q2 baseline study included three distinct threads 0.3.3,0.3.6,0.3.8 and we will develop enhancements for each. The priority will go to the study of implicit nonlinear edge localized modes and the physical goals are described in 0.3.3. This particular study is composed of three problem instances: a medium and large linear problem, and a nonlinear problem. The 'medium linear' problem has a 40x65 mesh of biquintic finite elements to represent the spatial variations of 8 physical quantities for a single Fourier

component, 500,000 complex degrees of freedom. The 'large linear' problem has better poloidal resolution with an 80x128 mesh of biquintic elements. Improvements in poloidal resolution are needed to run conditions with more realistic dimensionless dissipation parameters. The third problem has the poloidal resolution of the 'medium linear' problem and includes 11 coupled Fourier harmonics, $0 \leq n \leq 10$. As an example of the measured results, the large linear problem was executed on the Cray XK6 at NERSC utilizing 128 PEs, retiring 4.24×10^{13} instructions and 4.02×10^{11} floating point operations in 111.21 seconds. The goal is to improve the efficiency of the solves. Software issues are presented in the appendix 0.7.3.

- What is optimization through machine-learned chemistry? VASP and most other computational quantum mechanics codes have two nested optimization loops. The inner loop achieves convergence of the electronic problem (the Kohn-Sham equations) for a given static configuration of the ions. At the end of each electronic convergence, forces on the atoms are calculated from the converged wave functions, and the ions are displaced in the direction of the forces according to some simple spring model. Executing both loops (ionic and electronic convergence) ultimately leads to the atomic positions that provide the lowest energy within the symmetry constraints specified in the input cell. This ionic relaxation is remarkably inefficient as it is simply treated as a mathematical problem without any chemical input. For example, stiff bonds, soft bonds, atomic units, etc. are all treated in exactly the same way. This can lead to a very large number of iterations when systems with multiple bonding energy scales are studied. Such mixed energy scales is the typical case, rather than the exception. Many materials have strongly bonded atomic groups (e.g. the SiO₄ group in silicates a common group of minerals -) which can loosely rotate or displace. If VASP were aware of such chemical information, then it could treat such hard and soft degrees of freedom more efficiently. We propose to use machine learning techniques to mine chemical bonding knowledge from the tens of thousands of relaxations that have already been executed in the Materials Project (www.materialsproject.org) and use that knowledge to a) pre-optimize any input cell to VASP, and b) transform the Cartesian coordinate system used in VASP into a more relevant coordinate that reflects the variance these coordinates have on the total energy. The idea is that we use local learning techniques in each ionic iteration to learn about the important and less important degrees of freedom in a system, as well as global learning across the tens of thousands of systems on which VASP is being run. The benefits of such machine learning optimizations will be substantial. They will be additive to whatever direct numerical optimization will be performed on VASP. Additionally, ionic relaxation improvements are in principle code-independent and can also be ported to other common quantum mechanical codes such as ABINIT, Wien2K and QuantumEspresso. Another problem that may be pursued in the context of VASP computations is the introduction of a k-space parallelization algorithm in the static solver of the self-consistent iteration. Solvers that exploit homogeneities in the problem geometry and can execute each k_z value independently within

a self-consistent iteration. This amounts to forming independent parallel work groups, each of which simultaneously numerically diagonalizes in parallel a reduced dimension problem each self-consistent iteration. The special case achieves nearly perfect strong scaling when computing all the k -dependent diagonalizations at once versus in sequence. Furthermore, the cost of a single diagonalization is reduced. We expect to test this parallelization scheme at both the OLCF and NERSC as there is a fairly large user base that will benefit from these improvements.

- Representative problem instances from the ICSD (Inorganic Crystal Structures Database) have been defined for optimization. The study we have undertaken aims to improve the throughput of computing independent physical systems that depend on VASP computations. For the compounds selected in Q2, a set of base thermal, mechanical, and spectroscopic properties are calculated. In order to cover a representative span of these inputs a variety of band-gaps and ionic-step convergence properties were chosen. The numerical problem size is determined primarily by the atom count (electron count), the plane wave cutoff, and the number of k -points within the Brillouin zone. The test problems consist of relaxing the atoms, volume, and shape of the supercell, and determining the cohesive energy of the resultant fully relaxed supercell. The performance details for the sample set can be found starting in Table 8 and the inputs and build process 0.7.2. We will exploit a k -space parallel discretization to exploit strong scaling features of these problems.
- QMCPACK implements continuum quantum Monte Carlo (QMC) methods for predicting the properties of matter from first principles. By solving the many-body Schrödinger equation through stochastic projection, QMC achieves greater accuracy than other electronic structure methods such as density functional theory, and much better scalability than quantum chemical methods, enabling scientific discovery across physics, materials science and chemistry. QMCPACK is an open-source QMC package designed for large-scale clusters of multi-core SMPs and GPUs. The multiple forms of parallelism afforded by QMC algorithms make it an ideal candidate for acceleration in the many-core paradigm. The accuracy and efficiency of a QMC calculation is determined by the many-body trial wave function. Various forms of trial wave functions are used to capture the correlation at the minimum computational cost. The core of computation involves evaluating single-particle orbitals for electrons (Fermions), Slater determinant updates, and pair-wise terms for the correlation functions. Communication over multiple nodes is needed to update global quantities, e.g., the trial energy, and to perform load balancing as the population evolves with the MC projections. The multiplicity of the walkers of a population provides the natural unit for parallelism and the walkers are distributed over MPI nodes. Each task exploits light-weight threads to handle the walkers on a node and nested loops over the particles, electrons and ions of a system. This multi-level parallelism currently allows QMCPACK to scale to 200K cores on Jaguar at ORNL and to 200 GPUs at a sustained 90% parallel efficiency. We are pursuing a balanced use of hy-

brid multi-core host + GPU architectures. Today, the task-based parallelization scheme only makes effective use of a single processor core on the host. This process manages the GPU computation and communicates through MPI with other processes. The problem will be to study the phases of Ti and TiO_2 . The plan is to balance the assignment of multiple walkers on CPU cores and the GPU, and carry out the usual QMC algorithms. One challenge will be to restructure the management of an anonymous buffer for each walker that is used for the data transfer between CPU and GPU and between MPI nodes.

- The diffusion Monte Carlo calculation to reach < 1 mHa error in the total energy of a fixed normal electronic system with given input single-particle orbitals is to be optimized. With trial wavefunctions computed, several problem sizes can be studied each denoted by $gr(N_{xy} \times N_{xy} \times N_z)$ with $N_{xy} = 3, 4, 6, N_z = 1, 2$. In Q2, a graphite system consisting of 64 carbon atoms (256 electrons), $gr(4 \times 4 \times 1)$ was baselined. Table 13, displays results of timings for this system on hybrid software and hardware in a strong scaling study fixed at 128 walkers per node.

Project Objectives, Methods

Project Objectives SC GG 3.1/2.5.2 Improve computational science capabilities, defined as the average annual percentage increase in the computational effectiveness (either by simulating the same problem in less time or simulating a larger problem in the same time) of a subset of application codes. FY12 performance metric: efficiency measure is $\geq^{**\%}$.

The effort to improve computational science capabilities is a year long effort requiring quarterly updates. The quarterly sequence of tasks for exercising this software metric are summarized here.

Quarter One (Q1) Tasks (deadline: December 31). Identify a subset of candidate applications (scientific software tools) to be investigated on DOE SC supercomputers. Management (at DOE SC and national laboratories) decides upon a short list of applications and computing platforms to be exercised. The Advanced Scientific Computing Advisory Committee (ASCAC) approves or rejects the list. The Q1 milestone is satisfied when a short list of target applications and machines (supercomputers) is approved.

Quarter Two (Q2) Tasks (deadline: March 31). Problems that each chosen application will simulate on the target machines are determined. The science capability (simulation result) and computational performance of the implementation are benchmarked and baselined (recorded) on the target machines for the defined problems and problem instances. The Q2 milestone is satisfied when benchmark data is collected and explained. If an application is striving to achieve a new science result in addition to demonstrating improved performance, then providing a detailed discussion of the existing capability, why it is insufficient and how the challenges / deficiencies are being addressed satisfy the Q2 milestone.

Quarter Three (Q3) Tasks (deadline: June 30). The application software (its models, algorithms, and implementation) is enhanced for efficiency, scalability, science capability, etc. The Q3 milestone is satisfied when the status of each application is reported at the Q3 deadline. Corrections to Q2 problem statements are normally submitted at this time.

Quarter Four (Q4) Tasks (deadline: September 30). Enhancements to the application software continue as in Q3. The enhancements are stated and demonstrated on the machines used to generate the Q2 baseline information. A comparative analysis of the Q2 and Q4 data is summarized and reported. The Q4 milestone is satisfied if the enhancements made to the application software are in accordance with the efficiency measure as defined in Q2 (run-time efficiency, scalability, or new result).

Method and Challenges It is not possible to provide a comprehensive description of the variety of methods employed and challenges encountered in scientific software applications. However, a the starting point for a majority of DOEs software applications is formulating the problem as a system of coupled, partial differential equations (PDEs), we quickly comment on the methods in common use by application developers. Techniques to solve PDEs depend on problem geometry and dimensionality, coordinate representation, order of the equations, boundary conditions, spatial and temporal scales, and so on. Most PDEs can be rewritten in differential algebraic or integral form, and are usually rewritten yet again to linearize the analysis. Approximations of analytic methods like separation of variables, Greens function techniques, or variational techniques like Rayleigh-Ritz-Galerkin methods that lead to non-linear optimization or eigenvalue problems are well studied and there is a broad range in the sophistication and implementation of algorithms and related data-structures in this space. Techniques for solving PDEs include structured and unstructured lattice methods within some function basis representation and some finite difference method to provide the calculus on a grid (or many grids), discrete integral transform methods, linear algebra factorizations and iterative schemes, stochastic methods such as Monte Carlo / Metropolis sampling, particle-in-cell (PIC) or Molecular Dynamics (MD) techniques, etc. Multi-level solvers process successive levels sequentially limiting subproblem parallelism. There exist block iterative smoothing techniques that can improve performance and deliver a coarsening that results in fewer levels requiring synchronization. Time integration techniques may be explicit or implicit. Runge-Kutta and operator splitting are typical methods but evaluation depends on the representation, accuracy, and tends to evolve in sequential steps. We focus on trade-offs between convergence rates, data structure and layout, memory demands, communication load and synchronization requirements and scaling features for various techniques. As an example, in FY11 a realistic Si double gate ultra thin body transistor configuration for several important features was studied. The software was not originally written to describe this problem and as such a non-optimal unit cell was being employed during the construction of matrix elements. Indeed, 3D nanowires are not periodic rather are confined in the lateral component. We decided to work with the primitive unit cell of UTB transistors instead of a more convenient, but larger unit cell that enables the exploitation of a physical symmetry that imposes k-dependence on transmission observables, the density of states, etc. The result of this change enables each sub-matrix in the Hamiltonian to be generated independently with low communication overheads. Further, a sparse pattern of the matrix sum emerges and can be stored to increase the initialization speed during each iteration. The Si UTB transistor problem including electron-phonon scattering was executed in both Q2 and Q4 until a fixed convergence between the Green's Functions and self-energies was achieved. The difference in costs was 151,467 CPU Hrs (Q2) - 69,817 CPU Hrs (Q4) = 81,650 CPU Hrs and leads to greater than a factor of two enhancement.

Programming methods in use by developers also play a critical role in computational science delivering the interface between humans and computing hardware. Software developers find themselves attempting to balance programming styles and patterns that lead to efficient code on a particular

architecture, with the need to enable portability across different platforms and processor types. The simple picture of the entire system that often inspires algorithm design is not clear. Many programmers are using evolutionary hybrid programming models such as MPI / OpenMP / PGAS. Several interesting software efforts are laying the foundation for testing new disruptive technologies. OpenCL supports a uniform programming environment composed of processing elements, compute units and devices, and targets heterogeneous systems. NVIDIA's CUDA C allows developers to write programs that will execute concurrently on two different platforms: a host system with one or more CPUs and one or more devices, i.e. CUDA-enabled NVIDIA GPUs. OpenACC, based on AMD's Accelerator research, relies on the notion that directive-based compilation can deliver the performance of accelerators for appropriate algorithm constructions. PGI and Cray compilers will be the first to support these directives. Intel claims that their MIC (Many Integrated Core) products and the KNC (Knights Corner) accelerator can be effectively utilized with the same OpenMP constructions used to target the Intel Xeon chip architectures. All these developments target compute node structures, not the perspective of the entire system that application developers tend to employ, nevertheless open an incremental path for moving legacy applications to multi-core and hybrid many-core systems. In FY11, the LAMMPS molecular dynamics software package was evaluated to test enhancements that exploit a hybrid node structure composed of the host and GPU. The user-cuda package aimed at balanced ratio of processor cores to GPUs was tested, and the gpu package aimed at hybrid nodes coupling multi-core hosts to GPUs. In both cases, the spatial decomposition was with MPI between CPU cores and the force (atom) decomposition executed on individual GPUs. We compared speed-ups in different scaling modes w/ increased numbers of time steps and atoms between the MPI and MPI + user-cuda , gpu versions of the algorithms for embedded atomic interaction of a Cu cluster and the Lennard-Jones(12,6) interaction for an atomic melting problem. The results of the gpu package obtained on the Dirac cluster at NERSC can be seen in Figure 1.

Programmers have to re-invent hit-or-miss strategies employed in code optimization for more sophisticated multicore CPUs. We focus on the following basic set of techniques and are still experimenting to gain control given the increased parallelism and abstracted programming models:

- non-temporal writes, ie dont cache the data writes since it wont be used again soon (i.e. n-tuple initialization) avoids reading cache line before write, avoids wasteful occupation of cache line and time for write (memset()); does not evict useful data sfence() compiler set barriers
- loop unrolling , transposing matrices
- vectorization, 2,4,8 elements computed at the same time (SIMD) w/ multi-media extensions to ISA

- reordering elements so that elements that are used together are stored together -pack CL gaps w/ usable data (i.e. try to access structure elements in the order they are defined in the structure)
- stack alignment, as the compiler generates code it actively aligns the stack inserting gaps where needed ... is not necessarily optimal -if statically defined arrays, there are tools that can improve the alignment; separating n-tuples may increase code complexity but improve performance
- function inlining, may enable compiler or hand -tuned instruction pipeline optimization (ie dead code elimination or value range propagation) ; especially true if a function is called only once
- prefetching, hardware, tries to predict cache misses -with 4K page sizes this is a hard problem and costly penalty if not well predicted; software (`void __mm_prefetch(void *p, enum __mm_hint h) -__MM_HINT_NTA` -when data is evicted from L1d -dont write it to higher levels)

We are now in a position to focus on other indicators of performance or scaling bottlenecks such as the notion that performance and power consumption are determined by the rate of issuing instructions and the ratio of global memory transactions to the aggregated number of retired instructions can be directly tested through measurement. The reader is urged to contact the PI for more details.

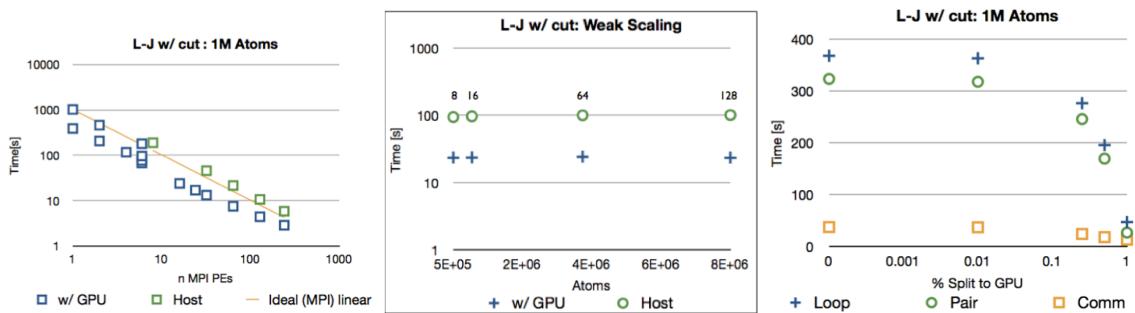


Figure 1: The gpu package of LAMMPS written by Michael Smith at ORNL was tested on the Dirac cluster at NERSC. The enhancements delivered a greater than two scaling result in both the weak and strong scaling modes. The plot on the right indicates the speed-up in the node gained when an increasing fraction of the force computation is passed to the GPU for execution.

Heterogeneous or hybrid hardware, communication synchronizations, I/O operations, methods that are inherently imbalanced such as PIC or AMR, various data and functional domain decompositions, thread affinity to hardware cores, fault or error recovery mechanisms, etc., are examples of application algorithm features that impose computational imbalances. We investigate what code

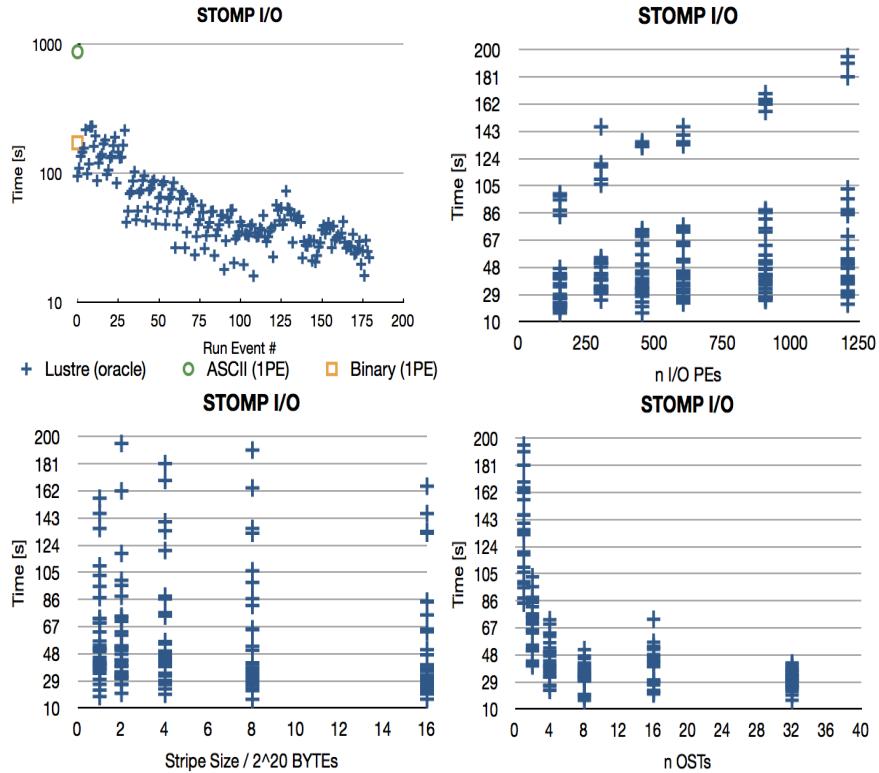


Figure 2: Results of new STOMP I/O Lustre-based I/O algorithm parameter study.

Lustre-based STOMP I/O			
Time[s]	Stripe Size [B]	n OSTs	n I/O PEs
16.064482	8388608	8	151
16.156173	16777216	32	453
18.029697	1048576	8	151
19.627054	4194304	8	151
19.776406	16777216	32	151

STOMP ASCII Unformatted I/O			
875.422139			1

STOMP Binary Formatted I/O			
172.573337			1

Table 1: Consistently fast parameter sets employing the Lustre explicit STOMP I/O algorithm. Only parameter sets that completed *all* the Q2 problem I/O in less than 20s are printed. The stand-alone MPI test codes that exactly mimick STOMP's GA-based I/O are compared.

regions will contain load imbalances and what the user would like to do about it. Also, we anticipate a time when applications will exploit the fact that there is a correlation between the clock frequency and achievable performance such that raising or decreasing the clock speed raises or lowers the theoretical limit of performing certain tasks. Intel has demonstrated that power can be saved by finishing CPU-intensive tasks faster and going to an idle state for the remaining duration, and other research has shown that maximizing certain CPU and memory settings delivers the best possible performance on several common, relevant application kernels including matrix-multiplication and Fast Fourier Transforms (FFTs). We have now started an extension of our analysis to estimate the benefits from changing the available power for short durations to enable increased performance or to arrange a smooth entry into synchronization points and the execution of certain critical kernels.

I/O events play a prominent role in many applications and often cannot be organized in a manner that hides the time penalties involved with overlapping computations. A common pattern for check points and data writes is that generated information that is domain decomposed over a set of distinct processes has to be collected to a smaller set of processes that will actually execute the I/O task leaving the remainder processes with nothing to do. Similarly, a smaller set of processes may be tasked to read and distribute a data set to a larger number of distinct processes according to some data decomposition, such as in a restart or initialization phase. It is often the case that these distribution phases are done by moving blocks of data through a collective operation to the global set of processes, followed by a parsing phase where only processes that own a segment of data engage in copying data to work buffers. We often exploit the data movement patterns and process behaviors of the collection or distribution phases of various I/O algorithms present in our applications to create improved algorithms better positioned for scalability as well as effective use of the high-performance file system software on the LCF systems. Consider an example from the FY11 application STOMP (Subsurface Transport Over Multiple Phases) on the Cray XT5. The problem is a physical cartesian lattice of points distributed over some virtual grid of processes. A number of fields of different data types are written for plotting and checkpoint restart. Our analysis suggested that a buffered, collective I/O algorithm that targeted Lustre could greatly speed up the computation. We wrote software that forms modulo classes from the global MPI communicator over the number of I/O groups and delivers 2 levels of parallelism: over the fields, and spatial decomposition w/ correct indexing. An oracle code was written that tuned parameters set in the file system such as the number of OSTs and the stripe size to the problem parameters. Figure 2 depicts the reproducible gains and the new I/O parameters with the new algorithm reduced the I/O overhead, a dominant cost, by 54X in the Q4 simulation.

Computational Science Capabilities, Q2 Metric Status

0.1 Drekar: Next Generation CFD Simulations of Turbulent Fluid Flow and Heat Transfer in PWRs

0.1.1 Introduction

0.1.2 Background and Motivation

The Consortium for Advanced Simulation of Light Water Reactors (CASL) is the name of a new U.S. Department of Energy Innovation Hub that is investing in the development of a “virtual reactor toolkit” that incorporates science-based models, state-of-the-art numerical methods, modern computational science, modern software engineering, and uncertainty quantification and validation against currently operating pressurized water reactors. Under CASL there are five focus areas. One of these, Thermal Hydraulics Methods (THM) is tasked with developing computational fluid dynamics (CFD), multi-phase computational fluid dynamics (MCFD) and thermal-hydraulics (TH) tools that will be applied to challenge problems identified by industrial and academic partners as requiring high fidelity solution techniques in order to solve.

One such challenge problem deals with grid-to-rod-fretting (GTRF). This problem is characterized by flow induced rod vibrations that cause deterioration of the rod cladding material and support grids at points of contact. The vibrations are due to turbulent flow generated at the core inlet and by rod bundle support grid mixing vanes. Turbulence is deliberately generated to enhance heat transfer and prevent localized hot spots from occurring. This problem is inherently three-dimensional and unsteady. High fidelity turbulence modeling will lead to better understanding of rod excitation phenomena and has the potential to improve clad time-to-failure, enhance heat transfer and improve overall reactor core performance.

Another challenge problem that will receive increased attention in the near future is corrosion related unidentified deposits (CRUD) formation at “hot spots” on active fuel rods. The physics of CRUD includes species transport and reaction, thermal transport and turbulent mixing.

Ultimately, full-scale simulations of the entire reactor cores including inlet/outlet characterization will be performed. Included in these will be coupled fluid/structure interaction, thermal-hydraulics, radiation transport, species transport and reaction simulations.

These physical phenomena are represented by conservation laws and described mathematically

by systems of partial differential equations. These PDEs are nonlinear, time dependent, three-dimensional and exhibit large ranges in time and length scales. In addition, the nuclear reactors that create these flows are very geometrically complex. The range in length and time scales inherent in fluid motion increases dramatically as the Reynolds number increases. This renders the PDEs describing fluid motion, namely the Navier-Stokes equations intractable for the kinds of flows encountered in nuclear reactors. This necessitates the need to employ some type of filtering such as Reynolds time averaging resulting in the Reynolds Average Navier-Stokes equations (RANS) or spatial filtering resulting in Large Eddy simulation (LES).

0.1.3 Capability Overview

Our approach to solving these massive, complex systems draws upon over a decade and a half of work on scalable, robust, accurate, and efficient unstructured finite element (FE) methods employing Newton-Krylov solution techniques with advanced domain decomposition, physics-based, approximate block factorization, and multi-level preconditioning methods. These techniques allow for the development of robust and efficient fully implicit time integration, and direct-to-steady-state solvers, using strongly coupled iterative solution methods for the large-scale systems that are produced by discretization of the governing PDEs.

Initially, code development has been focused on solving the GTRF challenge problem. An incompressible fluid is assumed. Both RANS and LES turbulence models are employed. Thermal transport is modeled through a temperature equation. Conjugate heat transfer is solved through a heterogeneous system of equations where the heat equation is solved in the solid regions simultaneously with the Navier-Stokes and temperature equations solved in the fluid regions. This heterogeneous system of equations is solved in a tightly coupled fashion.

The code infrastructure is very general and supports a much richer variety, of systems of equations, including low Mach number flow equations, and the capability to solve multiple momentum, energy and species equations simultaneously.

Leading up to full-scale, various sub-scale rod bundle assembly problems serving as prototypes will be solved and analyzed. Separate fluid and structural dynamic simulations will be conducted where the rod excitation predicted by the fluid simulations will be transferred to the structural code through surface boundary conditions.

0.1.4 Science Driver for Metric Problem

The science driver for the metric problem is the simulation of turbulent flow in a nuclear reactor core for the GTRF problem as described in section 0.1.2. While our code has been run in the range of 140K cores on an IBM Blue Bene P, the large scale results have been primarily for capability demonstration and associated with steady-state computations [?]. In this initial optimization effort the weak scaling and the CPU time performance of the application of the Krylov solver iteration and the AMG V-cycle preconditioner kernel is considered [?]. This kernel is just one of the most significant CPU time intensive aspects of the Drekar code. However in the case of time-dependent turbulence the CPU time is dominated by the construction of the AMG V-cycle preconditioner (see Table 6). This routine involves the construction of the projection matrices (prolongation and restriction), the construction of the operator hierarchy by projection, and construction and solution of the coarsest grid direct LU factorization. These CPU intensive kernels essentially are determining the overall time to solution of the CASL relevant transient turbulence modeling efforts.

The work we will perform on the ONRL Jaguar Cary XK6 this year will be to provide a *production* capability at a scale of 131K cores to support CASL work on the reactor core model. We are not using the full machine since the remaining cores are expected to be used by other application codes that are strongly coupled to Drekar in the virtual reactor suite being developed by CASL. The goal of the Drekar Joule metric is at least a factor of 2 speedup for the swirling jet test problem being run at 1.3 billion unknowns on 1313072 cores of Cray XK6 jaguar. We will work to improve the problem setup, equation set assembly, the preconditioner construction, and solution execution times.

The Joule metric problem chosen is a called a “swirling jet”. This is a non-proprietary problem in the Drekar test suite that mimics the fluid flow in the pin assemblies of a nuclear reactor just downstream of the mixing vane and central tube that is shown in Figure 3. Essentially we have taken the general characteristics of the fluid flow downstream of the mixing vane that produces a swirling jet and idealized this flow (See Figure 4). This problem provides the rich physics associated with turbulent CFD modeling yet eliminates the practical difficulties encountered when working with proprietary meshes with complex spatial features.

The swirling jet problem is a high Reynolds number flow ($1.0e+6$) enclosed in a rectangular domain. The initial condition consists of (1) a uniform flow with velocity $\mathbf{U} = (0, 0, 10)$ in the region outside of the unit cylinder centered at $(x_0, y_0) = (0, 0)$ and extended through the domain in the z-direction and inside of the rectangular domain of $[x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}] = [-3, 3] \times [-3, 3] \times [0, 18]$, and (2) a swirling flow inside the unit cylinder with velocity $\mathbf{U} = (-40y, 40x, 20)$. In the transient problem a turbulent flow profile develops as the twisting flow moves down the length of the domain. Periodic boundary conditions are applied to the side walls. No stress bound-

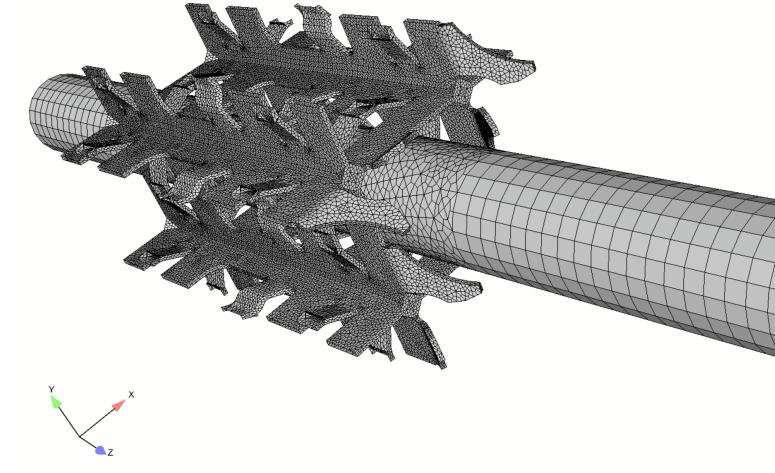


Figure 3: Images of problem geometry. Right image is the coarse 600K CUBIT HEX element mesh for mixing vane and center tube.

ary conditions are applied at the exit. A representative solution that plots a constant velocity iso-surface is shown in Figure 4. The transient SARANS turbulence model is used. For the Q2 metrics, the simulation is run for 0.1 seconds and resulted in 62 time steps.

The main utility of this problem is the elimination of the complex spatial features present in the mixing vane of the rod assemblies. Adequately meshing such a geometry involves expert meshing assistance (e.g. it is not an automated process) and multiple trial-and-error calculations to evaluate error in the meshes, especially when using turbulence. Given the time scales for the joule metric and current resources, producing multiple meshes of varying cell densities was prohibitively expensive (a problem the CASL project is currently addressing). By using the swirling jet problem, we can immediately control the grid peclet numbers. Additionally since the mesh is so simple we can cut out the trail-and-error loop with the meshing team and can autogenerate the mesh inline. This allows for scaling studies with fine grained control over the mesh and, most importantly, allows us to focus on improving the solution algorithms with fast turn around on jaguar. Any improvements on this model directly translate into improvements on the actual proprietary pin assembly simulations used in CASL.

Finally we note that since this problem is nonproprietary, the simulations are easier to deal with in the current computing environment in terms of data protection and export control. Additionally,

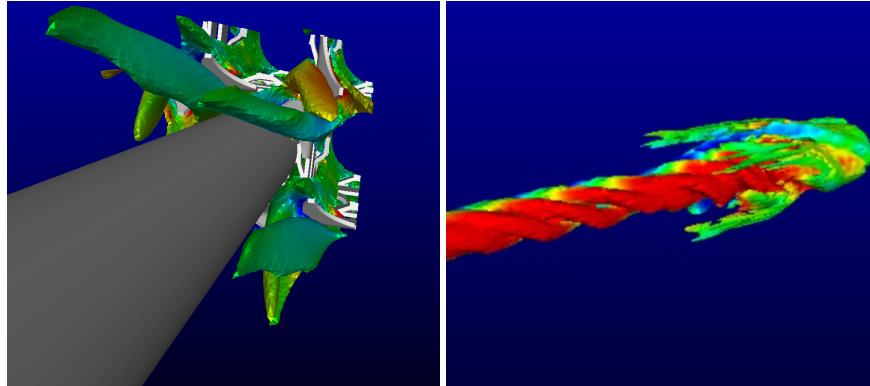


Figure 4: Left: Image of constant velocity surface swirling flow generated by mixing vane. Right: Image of constant velocity surface in idealized "swirling jet" problem.

the simulation results do not have to be approved by multiple partner institutions before release as opposed to using the actual reactor geometries.

0.1.5 Software Description: Model and Algorithm

This section gives a brief description the physics model and the solution methods used by Drekar. Given that Drekar is a general multiphysics capability, there are a number of physics implementations and solution algorithms in the code that are not required for this particular joule metric problem. This summary only discusses the equations relevant to the joule metric run and leaves out multiple alternative implementations for turbulent CFD models (i.e. $k - \varepsilon$, large eddy simulation), reacting flows, and magnetohydrodynamics physics. See the Drekar Theory manual [?] for more details.

Governing Equations

The equations governing fluid motion are the Navier-Stokes equations. These equations are listed in Table 2. They represent the conservation of mass and momentum. To these are added an energy equation describing the transport of heat. These equations are written in “residual” form which helps facilitate presenting the discretization of these equations via the finite element method.

Governing Equations	
Continuity	$R_\rho = \frac{\partial \rho}{\partial t} + \rho \nabla \cdot \mathbf{u}$
Momentum	$R_m = \rho \frac{\partial \mathbf{u}}{\partial t} + \rho \mathbf{u} \cdot \nabla \mathbf{u} - \nabla \cdot \mathbf{T} - \rho \mathbf{f}_i$
Energy	$R_T = \rho C_p \frac{\partial T}{\partial t} + \rho C_p \mathbf{u} \cdot \nabla T + \nabla \cdot \mathbf{q} + \nabla \cdot \mathbf{q}_r - \nabla \cdot \mathbf{u}$

Table 2: Navier Stokes Equations.

In these equations, ρ is the density, $\mathbf{u} = u_i$, $i = 1, 2, 3$ is the velocity vector with index i representing the Cartesian components in the (x, y, z) directions, T is the temperature and C_p is the specific heat at constant pressure.

\mathbf{T} is the stress tensor for a Newtonian fluid:

$$(1) \quad \mathbf{T} = -P\mathbf{I} + \boldsymbol{\tau} = -P\mathbf{I} + \mu_{eff}[\nabla \mathbf{u} + \nabla \mathbf{u}^T] - \frac{2}{3}\mu_{eff}(\nabla \bullet \mathbf{u})\mathbf{I}$$

where P is the isotropic hydrodynamic pressure, μ_{eff} is the effective dynamic viscosity that can have contributions from turbulence models (e.g., $\mu_{eff} = \mu + \mu_t$), and \mathbf{I} is the unity tensor.

The heat flux vector is given by the Fourier model:

$$(2) \quad \mathbf{q} = -\kappa \nabla T$$

where κ is the coefficient of thermal conductivity which can be specified by the specific heat capacity, dynamic viscosity and Prandtl number (Pr); $\kappa = \mu C_p / Pr$.

A radiation source, \mathbf{q}_r , also appears in the temperature equation. For low speed flows, $\rho \mathbf{f}_i$ would induce a buoyancy force through temperature gradients that have components orthogonal to the gravity vector.

In the case of low speed incompressible flow, the viscous dissipation flux term $\nabla \cdot (\boldsymbol{\tau} \cdot \mathbf{u})$ is written in non-conservative terms as;

$$\Phi = (\boldsymbol{\tau} : \nabla \mathbf{u}) = \frac{1}{2}\mu \|\nabla \mathbf{u} + \nabla \mathbf{u}^T\|^2 - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})^2$$

and is implemented as a source term. This term is sometimes omitted from the equation.

In many cases, where the Reynolds number is sufficiently high, the flow can be characterized as being turbulent. One of the key features of turbulence that has implications to CFD is the vast

range of time and length scales that coexist in the flow field. In theory, the unsteady Navier-Stokes equations (2) can capture turbulence with no additional modelling assumptions. This is referred to as direct numerical simulation (DNS). However, in practice, the range of scale makes DNS prohibitively expensive and therefore some sort of averaging of the equations must be performed in order to make turbulence simulations tractable. In the process of averaging a closure relationship is created that essentially must be modeled. In this section we summarize the approach used in the Joule metric calculation called Reynolds averaged Navier-Stokes (RANS), where the equations of motion are time or ensemble averaged.

We now discuss the turnbulence model. Consider a time dependent field variable $\phi(\mathbf{x}, t)$. The time average or filter is defined as;

$$(3) \quad \bar{\phi}(\mathbf{x}) = \lim_{\Delta T \rightarrow \infty} \frac{1}{\Delta T} \int_{t_0}^{t_0 + \Delta T} \phi(\mathbf{x}, t') dt'$$

and the ensemble average is defined as

$$(4) \quad \bar{\phi}(\mathbf{x}, t) = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^{n=N} \phi(\mathbf{x}, t).$$

With this definition, any dependent variable can be decomposed into an average plus fluctuation,

$$(5) \quad \phi(\mathbf{x}, t) = \bar{\phi}(\mathbf{x}) + \phi'(\mathbf{x}, t)$$

and the following identities apply;

$$(6) \quad \begin{aligned} \overline{\phi'} &= 0 \\ \overline{\bar{\phi}} &= \bar{\phi} \\ \overline{\phi\psi} &= \bar{\phi}\bar{\psi} + \overline{\phi'\psi'} \end{aligned}$$

$$(7) \quad \overline{\phi'\phi'} \neq 0.$$

Note that the third identity creates the closure problem for nonlinear equations such as Navier-Stokes. If this filter is applied to the incompressible constant density single component continuity

and momentum equations, the resulting equations become;

$$(8) \quad \frac{\partial \rho}{\partial t} + \nabla \cdot \bar{\mathbf{u}} = 0$$

$$(9) \quad \rho \frac{\partial \bar{\mathbf{u}}}{\partial t} + \rho \bar{\mathbf{u}} \nabla \cdot \bar{\mathbf{u}} - \nabla \cdot \bar{\mathbf{T}} - \sum_k^{N_g} \rho \mathbf{g}_k = 0$$

$$(10) \quad \rho C_p \frac{\partial \bar{T}}{\partial t} + \rho C_p \bar{\mathbf{u}} \cdot \nabla \bar{T} + \nabla \cdot \bar{\mathbf{q}} - \nabla \cdot \bar{\Phi} = 0$$

(11)

where the stress tensor is;

$$(12) \quad \bar{\mathbf{T}} = -\bar{P}\mathbf{I} + \bar{\tau} + \tau_t = -\bar{P}\mathbf{I} + \bar{\mu}[\nabla \bar{\mathbf{u}} + \nabla \bar{\mathbf{u}}^T] - \frac{2}{3}\bar{\mu}(\nabla \cdot \bar{\mathbf{u}})\mathbf{I} - \rho \bar{\mathbf{u}}' \bar{\mathbf{u}}'$$

and it should be understood that $\mathbf{u}\mathbf{u}' = \mathbf{u} \otimes \mathbf{u} = u_i u_j$ represents the outer product of two vectors. The equations look very similar to the unfiltered equations with the exception of a new term call the Reynolds stress τ_t ,

$$(13) \quad \tau_t = -\rho \bar{\mathbf{u}}' \bar{\mathbf{u}}'.$$

This new apparent stress arises mathematically through the filter operation applied to the nonlinear advection term. It is responsible for the modification of the flow field through turbulent fluctuations. In terms of numerical simulation, it represents the “closure problem” in that additional information or equations are now required that describe τ_t in order to close the filtered system of equations making solutions possible.

A common approach is to use the Boussinesq approximation where the Reynolds stresses are related to the filtered stresses through the following;

$$(14) \quad \tau_t = -\rho \bar{\mathbf{u}}' \bar{\mathbf{u}}' \approx \mu_t [\nabla \bar{\mathbf{u}} + \nabla \bar{\mathbf{u}}^T] - \frac{2}{3}[\mu_t \nabla \cdot \bar{\mathbf{u}} + \rho k]\mathbf{I}$$

where μ_t is the turbulent or eddy viscosity and

$$(15) \quad k = \frac{1}{2} \bar{\mathbf{u}}' \cdot \bar{\mathbf{u}}' = \frac{1}{2} (\bar{u}' \bar{u}' + \bar{v}' \bar{v}' + \bar{w}' \bar{w}')$$

is the turbulent kinetic energy. It is convenient to redefine the pressure as the sum of the static pressure and the turbulent kinetic energy;

$$(16) \quad P = \bar{P} + \frac{2}{3}\rho k.$$

And now the closure problem reduces to the task of calculating the eddy viscosity.

The averaged temperature equation contains the turbulent heat closure term, $\rho\bar{\mathbf{u}'h'}$, which includes the fluctuating enthalpy $h' = C_p T'$ and is modeled using the Boussinesq analogy;

$$(17) \quad \mathbf{q}_t = -\rho\bar{\mathbf{u}'h'} \approx -\kappa_t \nabla \bar{T}$$

where a turbulent or eddy conductivity is defined as;

$$(18) \quad \kappa_t = \frac{\mu_t C_p}{Pr_t}$$

and Pr_t is the turbulent Prandtl number.

Similar to the assumption that lead to equation 12 the total heat flux combines the laminar and turbulent contributions; ($\bar{\mathbf{q}} = \mathbf{q} + \mathbf{q}_t$). In the simulation of turbulent flows, the viscosity coefficient in the viscous fluxes is replaced with the sum of the natural and eddy viscosity ($\mu_t + \bar{\mu}$) and the heat conductivity is replaced by the sum of natural and turbulent conductivity ($\kappa_t + \bar{\kappa}$) in equations (1, 2). The turbulent Prandtl number Pr_t is also specified.

For RANS models that do not model the turbulent kinetic energy such as the Spalart-Allmaras [?] model presented below, the terms that contain turbulent kinetic energy that appear in the Favre averaged momentum and energy equations are simply neglected. The assumption being made that it is negligible compared with the total energy.

Note that an additional closure term appears in the temperature equation due to filtering the dissipation term;

$$(19) \quad \bar{\Phi} = \overline{\tau : \nabla \mathbf{u}} = \bar{\tau} : \nabla \bar{\mathbf{u}} + \overline{\tau' : \nabla \mathbf{u}'}$$

The second term on the r.h.s. of equation 19 is currently being neglected.

The Spalart-Allmaras turbulence model equation [?] for an incompressible fluid is given by,

The turbulent viscosity is given by,

$$(20) \quad \mu_t = \rho \hat{v} f_{v1}.$$

Table 3: Spalart-Allmaras RANS Turbulence Model Equation.

$$R_{\hat{v}} = \frac{\partial \rho \hat{v}}{\partial t} + \rho \bar{\mathbf{u}} \nabla \cdot \hat{v} - \nabla \cdot \left(\rho \left(\frac{\bar{v} + \hat{v}}{\sigma} \right) \nabla \hat{v} \right) - C_{b1} \rho \hat{v} \hat{S} + C_{w1} f_w \rho \left(\frac{\hat{v}}{d} \right)^2 - \frac{C_{b2} \rho}{\sigma} (\nabla \hat{v} \cdot \nabla \hat{v})$$

Functions defining the source terms and non-conservative viscous terms in the model are listed below;

$$(21) \quad \begin{aligned} \chi &= \frac{\hat{v}}{\bar{v}} \\ f_{v1} &= \frac{\chi^3}{\chi^3 + C_{v1}^3}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, \quad f_w = g \left(\frac{1 + C_{w3}^6}{g^6 + C_{w3}^6} \right)^{1/6} \\ g &= r + C_{w2} (r^6 - r), \quad r = \frac{\hat{v}}{\hat{S} k^2 d^2} \\ \hat{S} &= (2\Omega\Omega)^{1/2} + \frac{\hat{v} f_{v2}}{k^2 d^2}, \quad \Omega = \frac{1}{2} (\nabla \bar{\mathbf{u}} - \nabla \bar{\mathbf{u}}^T) \end{aligned}$$

where Ω is the rotation tensor and the model constants are,

$$(22) \quad \begin{aligned} k &= 0.41 \\ C_{b1} &= 0.1355, \quad C_{b2} = 0.622 \\ \sigma &= 2/3 \\ C_{w1} &= \frac{C_{b1}}{k^2} + \frac{1 + C_{b2}}{\sigma}, \quad C_{w2} = 0.3, \quad C_{w3} = 2.0 \\ C_{v1} &= 7.1 \end{aligned}$$

k is von Karman's constant, and d appearing in the source terms represents the normal distance to the wall. At a wall $\mu_t = 0$, and therefore the boundary condition is $\hat{v} = 0$.

Discretization

In this section we very briefly describe the stabilized FE methods used for discretization of the Navier-Stokes equations with coupled transport. This formulation follows the work of Hughes *et al.* [?] and Tezduyar [?]. The governing PDEs for the flow and transport system are presented in Table 2 along with the appropriate auxiliary equations that may be required to compute turbulent transport effects (e.g. $k - \epsilon$ equations). The stabilized method allows equal order interpolation of velocity and pressure and also provides stabilization of the convection operators to limit oscillations due to high grid Reynolds and Peclet number effects. Our formulation is capable of handling incompressible, low Mach number, variable density (temperature and chemical species dependent), and low speed compressible flows. This flexibility is enabled by the fully-implicit time integration and fully-coupled nonlinear solver technology that is described in the section 0.1.5. In terms of the spatial discretization technology the most challenging aspects are associated with the incompressible limit. In this case a mixed Galerkin FE formulation requires discrete spaces that satisfy the Ladyzhenskaya-Babuska-Brezzi (LBB) condition; (see e.g., [?]). Among other things, this condition prevents the use of equal order finite element spaces, defined with respect to the same partition of the computational domain into finite elements. Linearization of the mixed nonlinear equations also leads to indefinite linear systems that are more difficult to solve by iterative methods. An additional difficulty is that the mixed Galerkin formulation is prone to instabilities for highly convected flows, even if the LBB condition is satisfied by the finite element spaces.

Consistently stabilized finite element methods ([?]; [?]) address these issues by using properly weighted residuals of the momentum equation to simultaneously relax the incompressibility constraint and add streamline-diffusion and nonlinear-discontinuity-capturing operators (DCO) to the weak equations. A significant added advantage of stabilization is that the linearized problems are real positive definite and therefore performance of iterative solvers can be improved [?]. The FE weak form for the governing PDEs (equation 2) are presented in equations 23, 24, 25 and 26. This system employs equal-order interpolation, which allows one to simplify the data structures of a parallel unstructured FE code and the linear algebra interface for iterative solution methods ([?]; [?]; [?]). To improve stability for highly convected flows we employ discontinuity-capturing operators that provide additional isotropic, or anisotropic crosswind, diffusion to supplement the natural streamline diffusion contribution of the GLS stabilization. The resulting finite element formulation decreases numerical oscillations and allows for stable and accurate finite element solutions when the cell Reynolds, Re_c , and thermal energy and mass transport Peclet numbers, Pe_c , are greater than one.

Momentum

(23)

$$F_{\mathbf{u}_i} = \int_{\Omega} R_{m_i} \Phi d\Omega + \sum_e \int_{\Omega^e} \rho \tau_m (\mathbf{u} \cdot \nabla \Phi) R_{m_i} d\Omega + \sum_e \int_{\Omega^e} \rho \tau_c [\nabla \Phi]_i R_P d\Omega + \int_{\Omega^e} v_{m,i} \nabla \Phi \cdot \mathbf{C} \nabla \mathbf{u}_i d\Omega$$

Continuity

$$(24) \quad F_P = \int_{\Omega} R_P \Phi d\Omega + \sum_e \int_{\Omega^e} \rho \tau_m \nabla \Phi \cdot \mathbf{R}_m d\Omega$$

Energy

$$(25) \quad F_E = \int_{\Omega} R_E \Phi d\Omega + \sum_e \int_{\Omega^e} \rho C_p \tau_E (\mathbf{u} \cdot \nabla \Phi) R_E d\Omega + \int_{\Omega^e} v_E \nabla \Phi \cdot \mathbf{C} \nabla E d\Omega$$

Spalart-Allmaras

$$(26) \quad F_{\hat{v}} = \int_{\Omega} R_{\hat{v}} \Phi d\Omega + \sum_e \int_{\Omega^e} \rho \tau_{\hat{v}} (\mathbf{u} \cdot \nabla \Phi) R_{\hat{v}} d\Omega + \int_{\Omega^e} v_{\hat{v}} \nabla \Phi \cdot \mathbf{C} \nabla \hat{v} d\Omega$$

In these equations the SUPG and the VMS coupling stability parameters (the τ 's) are functions of the fluid velocity, \mathbf{u} and the transport coefficients of the PDEs and are given in [?], [?], [?]. Here the last integral operator on the R.H.S. of the Momentum, Energy and Spalart-Allmaras weak equations is a cross-stream oriented tensor that adds diffusion in the cross-stream direction, orthogonal to the SUPG contribution. Here it should be pointed out that the DCO coefficients (e.g. v_E) are proportional to the PDE residual and are therefore represent a consistent modification of the equations that the exact solution satisfies.

The specific definition of the intrinsic-time-scale stability parameters are provided in Table 4 for momentum, continuity, thermal energy, and the Spalart-Allmaras RANS equation. The specific form of the intrinsic time scales τ 's are an adaptation of the quadratic form suggested by Shakib [?] for the Navier-Stokes equations with coupled scalar transport. It should be pointed out

Table 4: Intrinsic-time-scale stability parameters for transport equations.

Momentum	$\tau_m = \left(\left(\frac{\rho}{\Delta t} \right)^2 + \rho^2 \mathbf{u} \mathbf{G}_c \mathbf{u} + 12\mu^2 \ \mathbf{G}_c\ \right)^{-1/2}$
Continuity	$\tau_c = (8 \ \mathbf{G}_c\ \tau_m)^{-1}$
Energy	$\tau_T = \left(\left(\frac{\rho C_p}{\Delta t} \right)^2 + (\rho C_p)^2 \mathbf{u} \mathbf{G} \mathbf{u} + 12\lambda^2 \ \mathbf{G}_c\ \right)^{-1/2}$
SARANS	$\tau_{\hat{v}} = \left(\left(\frac{\rho}{\Delta t} \right)^2 + \rho^2 \mathbf{u} \mathbf{G} \mathbf{u} + 12(\mu_{eff})^2 \ \mathbf{G}_c\ \right)^{-1/2}$

that the multidimensional effect of convection is incorporated into the stability parameters by the use of the contravariant metric tensor, \mathbf{G}_c (Equation (27)), of the transformation from local element coordinates $\{\zeta_\alpha\}$ to physical coordinates $\{x_i\}$,

$$(27) \quad [\mathbf{G}_c]_{ij} = \frac{\partial \zeta_\alpha}{\partial x_i} \frac{\partial \zeta_\alpha}{\partial x_j}.$$

Shakib [?] considers the one dimensional limiting case of this multidimensional definition for the advection-diffusion equation and presents a comparison with the original SUPG technique.

The anisotropic discontinuity capturing operator defined for the scalar component of velocity, u_i , is;

$$(28) \quad v_{m,i} = \max\{0, \chi \frac{\text{diam}(\Omega^e) \mathbf{R}_{m,i}}{2\|\nabla \mathbf{u}_i\|_2} - \mu\},$$

and the cross-wind orthogonal projection, \mathbf{C} , to the velocity vector, \mathbf{u} , is given by,

$$(29) \quad \mathbf{C} = \mathbf{I} - \frac{\mathbf{u} \otimes \mathbf{u}}{\|\mathbf{u}\|_2^2}.$$

Similar definitions are used for each of the scalar components for which the discontinuity operators are defined. The constant, χ , is a user defined value for which we take $\chi = 1/2$ in all our computations and currently we approximate the $\text{diam}(\Omega^e)$ by the norm of the metric tensor, (i.e., $\text{diam}(\Omega^e) \approx 1/\sqrt{\|\mathbf{G}_c\|}$).

Solution Algorithms

In the previous section, the finite element method was used to form the semi-discrete PDEs in space. The temporal operators have yet to be discretized. For these operators, the method of lines is used (i.e., time integration is performed in a manner similar to the way ODEs are integrated).

For stiff (multiple-time-scale) PDE systems such as the low Mach number Navier-Stokes system with coupled transport, fully-implicit methods are an attractive choice that can often provide unconditionally-stable time integration techniques. These methods can be designed with various types of stability properties that allow robust integration of multiple-time-scale systems without the requirement to resolve the stiff modes of the system (which are not of interest since they do not control the accuracy of time integration). The Drekar code can employ variable order (Order = 1-5) L-stable BDF methods from Trilinos/Rhythmos as well as a midpoint rule, and various types of Runge-Kutta (RK) type methods. Currently the BDFn methods have been verified up to order 4. In the future the multistage RK methods will also be used since they have a self-restarting capability.

The result of a fully-implicit or direct-to-steady-state solution technique is the development of very large-scale, coupled highly nonlinear algebraic system(s) that must be solved. Therefore, these techniques place a heavy burden on both the nonlinear and linear solvers and require robust,

scalable, and efficient nonlinear solution methods. In the Drekar code, Newton-based iterative nonlinear solvers ([?]) are employed to solve the challenging nonlinear systems that result from this application. These solvers can exhibit quadratic convergence rates independently of the problem size when sufficiently robust linear solvers are available. For the latter, we employ Krylov iterative techniques. A Newton-Krylov (NK) method [?] is an implementation of Newton's method in which a Krylov iterative solution technique is used to approximately solve the linear systems, $\mathbf{J}_k \mathbf{s}_{k+1} = -\mathbf{F}_k$, that are generated at each step of Newton's method. In Drekar backtracking algorithms are used to improve the robustness of the NK nonlinear solver by scaling the Newton correction vector to ensure a sufficient reduction of the nonlinear residual before the step is accepted ([?]; [?]).

For efficiency, an inexact Newton method ([?]; [?]) is usually employed, whereby one approximately solves the linear systems generated in the Newton method by choosing a forcing term η_k and stopping the Krylov iteration when the inexact Newton condition,

$$(30) \quad \|\mathbf{F}_k + \mathbf{J}_k \mathbf{s}_{k+1}\| \leq \eta_{k+1} \|\mathbf{F}_k\|$$

is satisfied. In general, nonlinear residual information is used to determine the forcing η_{k+1} . Finally it should be noted that in the Drekar code, the Jacobian matrix, \mathbf{J}_k , that is used for the matrix-vector products in the Krylov solvers, and as the basis for computing the preconditioners is developed from automatic differentiation (AD) techniques. These methods are applied to the programmed functions representing the weak form residuals outlined in section 0.1.5 by employing the SACADO package from the Trilinos framework [?].

For the considered class of linear systems described above, convergence is only achieved with preconditioning due to ill-conditioning in the underlying matrix equations ([?]). The Drekar code uses as a base-level method a Schwarz domain decomposition (DD) preconditioner, where the basic idea is to decompose the computational domain Ω into overlapping sub-domains Ω_i and then assign each sub-domain to a different processor ([?]). One application of the algorithm consists of solving on sub-domains and then combining these local solutions to construct a global approximation throughout Ω . The i^{th} sub-domain problem is usually defined by enforcing homogeneous Dirichlet boundary conditions on the sub-domain boundary, $\partial\Omega_i$. In the minimal overlap case, the algebraic Schwarz method corresponds to block Jacobi where each block contains all degrees of freedom (DOFs) residing within a given sub-domain. Convergence is typically improved by introducing overlap, which can be done recursively. Incomplete factorization, ILU(k), is employed to approximate the solution of the local Dirichlet problems and avoid the large cost of direct factorization ([?]).

We note that the one-level preconditioner is black-box in that the overlapping sub-domain

matrices are constructed completely algebraically. A drawback of the one-level Schwarz method is its locality. A single application of the algorithm transfers information between neighboring sub-domains. This implies that many repeated applications are required to combine information across the entire domain. Thus, as the number of sub-domains increases, the convergence rate deteriorates for standard elliptic problems due to the lack of global coupling ([?]). The convergence rate also deteriorates as the number of unknowns per sub-domain increases when ILU(k) is used for a sub-domain solver. To improve algorithmic performance, coarse levels can be introduced to approximate global coupling ([?]; [?]; [?]) and produce a multilevel preconditioner. The use of a coarse mesh to accelerate the convergence of a one-level Schwarz preconditioner is similar in principle to the use of a sequence of coarser meshes in multigrid methods ([?]).

In the multilevel solution methods used in Drekar only algebraically generated coarse operators are considered. These are significantly easier to implement with a complicated unstructured mesh than geometric coarse grids ([?]; [?]; [?]). Most algebraic multigrid methods (AMG) associate a graph with the matrix system being solved. Graph vertices correspond to matrix rows for scalar PDEs, while for PDE systems it is natural to associate one vertex with each nodal block of unknowns, e.g., velocity, pressure, energy and chemical species at a particular grid point. In the Trilinos/ML solvers used in Drekar, METIS and ParMETIS ([?]) are used to group fine graph vertices into aggregates so that each aggregate effectively represents a coarse graph vertex. These graph-partitioning packages subdivide the matrix graph so that each partition has no more nodes than a user supplied parameter and that each partition is somewhat spherically shaped. This graph partitioning is then applied recursively until the user specified number of levels has been achieved. Once the coarse mesh is determined, grid transfer operators are constructed that correspond to piecewise constant interpolation on each aggregate. This initial grid transfer can then be improved by smoothing the corresponding basis functions ([?]; [?]). For the problems that Drekar solves both a non-smoothed (NSA) and Petrov-Galerkin smoothed aggregation (PGSA) algorithm, denoted as aggC are used. These methods are used from the ML multilevel package in Trilinos and described in [?].

0.1.6 Q2 Baseline Problem Results

The swirling jet problem was run on 131072 cores of jaguarpf. The timings results from the papi counters are summarized in Table 5

Additionally, Drekar contains a time monitor that reports timings around individual sections and aggregated sections of the code. A small subset of the timings are listed in Table 6 .

Table 5: PAPI counter results for Swirling Jet simulation.

Metric	Total	High	Low	Standard Deviation
PAPI_TOT_INS	6349855288448493501	69221860529979	13625403143974	3990132072630
PAPI_FP_INS	130486495829888641	1871564141064	920850330536	14567229513
PAPI_L2_DCM	3256263293410301	39518057746	22238158231	1005820612
PAPI_real_cyc	47040078070063			
PAPI_real_usec	21381853669			
PAPI_user_cyc	46499024000000			
PAPI_user_usec	21135920000			

Timer	Low	Mean	High	Per Call
Total Time	2.133e+04 (1)	2.136e+04 (1)	2.138e+04 (1)	2.136e+04 (1)
Solve Time	1.762e+04 (1)	1.762e+04 (1)	1.762e+04 (1)	1.762e+04 (1)
Nonlinear Solve	1.762e+04 (73)	1.762e+04 (73)	1.762e+04 (73)	241.4 (73)
Prec. Const.	1.243e+04 (331)	1.252e+04 (331)	1.254e+04 (331)	37.81 (331)
Linear Solve	2959 (331)	2967 (331)	2968 (331)	8.964 (331)
Jacobian Assembly	1785 (331)	1805 (331)	1889 (331)	5.452 (331)
Residual Assembly	328.2 (404)	328.5 (404)	329.2 (404)	0.813 (404)

Table 6: Time Monitor results for Swirling Jet simulation run for 0.1 seconds. Values are in seconds. Numbers in parentheses are the number of times the particular method was called. The Min, Mean and Max are values reported over the number of MPI processes (131072).

Note that there is a difference of approximately 3700 seconds from the total execution time and the time spent solving the problem (first two rows of Table 6). This represents the setup and shutdown phases of the code. While this is appreciable for the joule metric run out to 0.1 seconds, our actual runs are on the order of 5.0 seconds. So this setup time becomes less of a factor. While we will focus some effort on this area, it is not of primary concern. We will focus the Q3/Q4 efforts on improving the solution and assembly times.

From the timings in Table 6, it is apparent that the bulk of the time in the solution algorithm is spent in the preconditioner. A closer look at the single Newton iteration below gives the breakdown of timings for an average preconditioner phase.

```
*****
-- Status Test Results --
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
**.....OR Combination ->
**.....AND Combination ->
Converged....F-Norm = 4.510e-06 < 1.000e-03
          (Unscaled Two-Norm, Absolute Tolerance)
**.....WRMS-Norm = 2.858e+01 < 1
          (Min Step Size: 1.000e+00 >= 1)
**.....Number of Iterations = 2 < 10
**.....Finite Number Check (Two-Norm F) = Finite
*****
*****
-- Nonlinear Solver Step 2 --
||F|| = 4.510e-06 step = 1.000e+00 dx = 3.935e+02
*****
CALCULATING FORCING TERM
Method: Constant
Forcing Term: 0.001
-----
ML time information (seconds)           total      avg
1- Construction                      =    10803.8     41.7136
2- Preconditioner apply              =     1601.79
   a- first application(s) only     =     21.5186    0.0830833
   b- remaining applications        =     1580.27    0.0818409
3- Total time required by ML so far is 12405.6 seconds
   (constr + all applications)
-----
Using 'cg' for eigen-computations
***
*** ML_Epetra::MultiLevelPreconditioner
***
Matrix has 1343488000 rows and 1.81253e+11 nonzeros, distributed over 131072 process(es)
The linear system matrix is an Epetra_CrsMatrix
** Leaving column map of Main linear system matrix unchanged
Default values for 'SA'
Maximum number of levels = 10
Using increasing levels. Finest level = 0, coarsest level = 9
Number of applications of the ML cycle = 1
Number of PDE equations = 5
Aggregation threshold = 0
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
Max coarse size = 5000
R and P smoothing : P = (I-\omega A) P_t, R = P^T
R and P smoothing : \omega = 0/lambda_max
Null space type      = default (constants)
Null space dimension = 5
-----
ML_Gen_MultiLevelHierarchy (level 0) : Gen Restriction and Prolongator
-----
ML_Aggregate_Coarsen (level 0) begins
ML_Aggregate_CoarsenUncoupled : current level = 0
ML_Aggregate_CoarsenUncoupled : current eps = 0.000000e+00
Aggregation(UVB) : Total nonzeros = 864288768 (Nrows=1343488000)
Aggregation(UVB) : Amalgamated matrix done
Aggregation(UC) : Phase 0 - no. of bdry pts = 262144
Aggregation(UC) : Phase 1 - nodes aggregated = 267387904 (268697600)
Aggregation(UC) : Phase 1 - total aggregates = 14008704
Aggregation(UC_Phase2_3) : Phase 1 - nodes aggregated = 266343420
Aggregation(UC_Phase2_3) : Phase 1 - total aggregates = 14008704
Aggregation(UC_Phase2_3) : Phase 2a- additional aggregates = 0
Aggregation(UC_Phase2_3) : Phase 2 - total aggregates = 14008704
Aggregation(UC_Phase2_3) : Phase 2 - boundary nodes = 1310720
Aggregation(UC_Phase2_3) : Phase 3 - leftovers = 0 and singletons = 0

Prolongator/Restriction smoother (level 0) : damping = 0.000e+00 , sweeps = 1
Gen_Prolongator (level 0) : not smoothing prolongator
ML_Gen_MultilevelHierarchy: Projecting node coordinates from level 0 to level 1
ML_Gen_MultilevelHierarchy (level 1): repartitioning suppressed until level 2
-----
ML_Gen_MultiLevelHierarchy (level 1) : Gen Restriction and Prolongator
-----
ML_Aggregate_Coarsen (level 1) begins
ML_Aggregate_CoarsenUncoupled : current level = 1
ML_Aggregate_CoarsenUncoupled : current eps = 0.000000e+00
Aggregation(UVB) : Total nonzeros = 471743424 (Nrows=70043520)
Aggregation(UVB) : Amalgamated matrix done
Aggregation(UC) : Phase 0 - no. of bdry pts = 0
Aggregation(UC) : Phase 1 - nodes aggregated = 6553600 (14008704)
Aggregation(UC) : Phase 1 - total aggregates = 524288
Aggregation(UC_Phase2_3) : Phase 1 - nodes aggregated = 6553600
Aggregation(UC_Phase2_3) : Phase 1 - total aggregates = 524288
Aggregation(UC_Phase2_3) : Phase 2a- additional aggregates = 384896
Aggregation(UC_Phase2_3) : Phase 2 - total aggregates = 909184
Aggregation(UC_Phase2_3) : Phase 2 - boundary nodes = 0
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
Aggregation(UC_Phase2_3) : Phase 3 - leftovers = 0 and singletons = 0
```

```
Prolongator/Restriction smoother (level 1) : damping = 0.000e+00 , sweeps = 1
Gen_Prolongator (level 1) : not smoothing prolongator
ML_Gen_MultilevelHierarchy: Projecting node coordinates from level 1 to level 2
Repartitioning (level 2): min rows per proc = 256
Repartitioning (level 2): largest max/min ratio = 1.200e+00
Repartitioning (level 2): max #rows (global) that fits on one proc = 5000
Repartitioning (level 2): #proc to use in repartitioning = 17757
Repartitioning using Zoltan 3.60
ZOLTAN Load balancing method = 3 (RCB)
Build configuration:
```

```
ZOLTAN_ID_TYPE: unsigned int (4 bytes)
ZOLTAN_GNO_TYPE: ssize_t, (8 bytes)
MPI_Datatype for ZOLTAN_ID_TYPE: MPI_UNSIGNED
MPI_Datatype for ZOLTAN_GNO_TYPE: MPI_LONG
Third party library: ParMetis version 3.2.0
Third party library: METIS 4.0 Copyright 1998, Regents of the University of Minnesota
```

```
ZOLTAN Parameter IMBALANCE_TOL[0] = 1.100000
ZOLTAN Parameter AUTO_MIGRATE = FALSE
ZOLTAN Parameter MIGRATE_ONLY_PROC_CHANGES = 1
ZOLTAN Parameter OBJ_WEIGHT_DIM = 1
ZOLTAN Parameter EDGE_WEIGHT_DIM = 0
ZOLTAN Parameter DEBUG_LEVEL = 1
ZOLTAN Parameter DEBUG_PROCESSOR = 0
ZOLTAN Parameter DETERMINISTIC = TRUE
ZOLTAN Parameter TIMER = 1 (wall)
ZOLTAN Parameter NUM_GID_ENTRIES = 1
ZOLTAN Parameter NUM_LID_ENTRIES = 0
ZOLTAN Parameter RETURN_LISTS = IMPORT AND EXPORT
ZOLTAN Parameter NUM_GLOBAL_PARTS = 17757
ZOLTAN Parameter NUM_LOCAL_PARTS = -1
ZOLTAN Parameter REMAP = 1
ZOLTAN Parameter SEED = 1992747517 (1992747517)
ZOLTAN Parameter LB_APPROACH = repartition
ZOLTAN Parameter RCB_OVERALLOC = 1.200000
ZOLTAN Parameter RCB_REUSE = 0
ZOLTAN Parameter CHECK_GEOM = 1
ZOLTAN Parameter RCB_OUTPUT_LEVEL = 0
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
ZOLTAN Parameter KEEP_CUTS = 0
ZOLTAN Parameter RCB_LOCK_DIRECTIONS = 0
ZOLTAN Parameter RCB_SET_DIRECTIONS = 0
ZOLTAN Parameter RCB_RECTILINEAR_BLOCKS = 0
ZOLTAN Parameter OBJ_WEIGHTS_COMPARABLE = 0
ZOLTAN Parameter RCB_MULTICRITERIA_NORM = 1
ZOLTAN Parameter RCB_MAX_ASPECT_RATIO = 10.000000
ZOLTAN Parameter AVERAGE_CUTS = 0
ZOLTAN Parameter RANDOM_PIVOTS = 0
ZOLTAN Parameter RCB_RECOMPUTE_BOX = 0
ZOLTAN Parameter REDUCE_DIMENSIONS = 0
ZOLTAN Parameter DEGENERATE_RATIO = 0.000000
ZOLTAN Parameter FINAL_OUTPUT = 0
ZOLTAN Parameter KEEP_CUTS = 0
ZOLTAN Parameter REDUCE_DIMENSIONS = 0
ZOLTAN Parameter DEGENERATE_RATIO = 10.000000
Zoltan (level 2) : time required = 5.949464e+00
-----
ML_Gen_MultiLevelHierarchy (level 2) : Gen Restriction and Prolongator
-----
ML_Aggregate_Coarsen (level 2) begins
ML_Aggregate_CoarsenUncoupled : current level = 2
ML_Aggregate_CoarsenUncoupled : current eps = 0.000000e+00
Aggregation(UVB) : Total nonzeros = 531425000 (Nrows=4545920)
Aggregation(UVB) : Amalgamated matrix done
Aggregation(UC) : Phase 0 - no. of bdry pts = 0
Aggregation(UC) : Phase 1 - nodes aggregated = 511014 (909184)
Aggregation(UC) : Phase 1 - total aggregates = 73921
Aggregation(UC_Phase2_3) : Phase 1 - nodes aggregated = 511014
Aggregation(UC_Phase2_3) : Phase 1 - total aggregates = 73921
Aggregation(UC_Phase2_3) : Phase 2a- additional aggregates = 18494
Aggregation(UC_Phase2_3) : Phase 2 - total aggregates = 92415
Aggregation(UC_Phase2_3) : Phase 2 - boundary nodes = 0
Aggregation(UC_Phase2_3) : Phase 3 - leftovers = 0 and singletons = 0

Prolongator/Restriction smoother (level 2) : damping = 0.000e+00 , sweeps = 1
Gen_Prolongator (level 2) : not smoothing prolongator
ML_Gen_MultilevelHierarchy: Projecting node coordinates from level 2 to level 3
Repartitioning (level 3): min rows per proc = 256
Repartitioning (level 3): largest max/min ratio = 1.200e+00
Repartitioning (level 3): max #rows (global) that fits on one proc = 5000
Repartitioning (level 3): #proc to use in repartitioning = 1804
Repartitioning using Zoltan 3.60
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

ZOLTAN Load balancing method = 3 (RCB)

Build configuration:

```
ZOLTAN_ID_TYPE: unsigned int (4 bytes)
ZOLTAN_GNO_TYPE: ssize_t, (8 bytes)
MPI_Datatype for ZOLTAN_ID_TYPE: MPI_UNSIGNED
MPI_Datatype for ZOLTAN_GNO_TYPE: MPI_LONG
Third party library: ParMetis version 3.2.0
Third party library: METIS 4.0 Copyright 1998, Regents of the University of Minnesota
```

```
ZOLTAN Parameter IMBALANCE_TOL[0] = 1.100000
ZOLTAN Parameter AUTO_MIGRATE = FALSE
ZOLTAN Parameter MIGRATE_ONLY_PROC_CHANGES = 1
ZOLTAN Parameter OBJ_WEIGHT_DIM = 1
ZOLTAN Parameter EDGE_WEIGHT_DIM = 0
ZOLTAN Parameter DEBUG_LEVEL = 1
ZOLTAN Parameter DEBUG_PROCESSOR = 0
ZOLTAN Parameter DETERMINISTIC = TRUE
ZOLTAN Parameter TIMER = 1 (wall)
ZOLTAN Parameter NUM_GID_ENTRIES = 1
ZOLTAN Parameter NUM_LID_ENTRIES = 0
ZOLTAN Parameter RETURN_LISTS = IMPORT AND EXPORT
ZOLTAN Parameter NUM_GLOBAL_PARTS = 1804
ZOLTAN Parameter NUM_LOCAL_PARTS = -1
ZOLTAN Parameter REMAP = 1
ZOLTAN Parameter SEED = 800870812 (800870812)
ZOLTAN Parameter LB_APPROACH = repartition
ZOLTAN Parameter RCB_OVERALLOC = 1.200000
ZOLTAN Parameter RCB_REUSE = 0
ZOLTAN Parameter CHECK_GEOM = 1
ZOLTAN Parameter RCB_OUTPUT_LEVEL = 0
ZOLTAN Parameter KEEP_CUTS = 0
ZOLTAN Parameter RCB_LOCK_DIRECTIONS = 0
ZOLTAN Parameter RCB_SET_DIRECTIONS = 0
ZOLTAN Parameter RCB_RECTILINEAR_BLOCKS = 0
ZOLTAN Parameter OBJ_WEIGHTS_COMPARABLE = 0
ZOLTAN Parameter RCB_MULTICRITERIA_NORM = 1
ZOLTAN Parameter RCB_MAX_ASPECT_RATIO = 10.000000
ZOLTAN Parameter AVERAGE_CUTS = 0
ZOLTAN Parameter RANDOM_PIVOTS = 0
ZOLTAN Parameter RCB_RECOMPUTE_BOX = 0
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
ZOLTAN Parameter REDUCE_DIMENSIONS = 0
ZOLTAN Parameter DEGENERATE_RATIO = 0.000000
ZOLTAN Parameter FINAL_OUTPUT = 0
ZOLTAN Parameter KEEP_CUTS = 0
ZOLTAN Parameter REDUCE_DIMENSIONS = 0
ZOLTAN Parameter DEGENERATE_RATIO = 10.000000
Zoltan (level 3) : time required = 9.880860e-01
-----
ML_Gen_MultiLevelHierarchy (level 3) : Gen Restriction and Prolongator
-----
ML_Aggregate_Coarsen (level 3) begins
ML_Aggregate_CoarsenUncoupled : current level = 3
ML_Aggregate_CoarsenUncoupled : current eps = 0.000000e+00
Aggregation(UVB) : Total nonzeros = 47854528 (Nrows=462075)
Aggregation(UVB) : Amalgamated matrix done
Aggregation(UC) : Phase 0 - no. of bdry pts = 0
Aggregation(UC) : Phase 1 - nodes aggregated = 55688 (92415)
Aggregation(UC) : Phase 1 - total aggregates = 7761
Aggregation(UC_Phase2_3) : Phase 1 - nodes aggregated = 55688
Aggregation(UC_Phase2_3) : Phase 1 - total aggregates = 7761
Aggregation(UC_Phase2_3) : Phase 2a- additional aggregates = 1439
Aggregation(UC_Phase2_3) : Phase 2 - total aggregates = 9205
Aggregation(UC_Phase2_3) : Phase 2 - boundary nodes = 0
Aggregation(UC_Phase2_3) : Phase 3 - leftovers = 5 and singletons = 5

Prolongator/Restriction smoother (level 3) : damping = 0.000e+00 , sweeps = 1
Gen_Prolongator (level 3) : not smoothing prolongator
ML_Gen_MultilevelHierarchy: Projecting node coordinates from level 3 to level 4
Repartitioning (level 4): min rows per proc = 256
Repartitioning (level 4): largest max/min ratio = 1.200e+00
Repartitioning (level 4): max #rows (global) that fits on one proc = 5000
Repartitioning (level 4): #proc to use in repartitioning = 179
Repartitioning using Zoltan 3.60
ZOLTAN Load balancing method = 3 (RCB)
Build configuration:

ZOLTAN_ID_TYPE: unsigned int (4 bytes)
ZOLTAN_GNO_TYPE: ssize_t, (8 bytes)
MPI_Datatype for ZOLTAN_ID_TYPE: MPI_UNSIGNED
MPI_Datatype for ZOLTAN_GNO_TYPE: MPI_LONG
Third party library: ParMetis version 3.2.0
Third party library: METIS 4.0 Copyright 1998, Regents of the University of Minnesota
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
ZOLTAN Parameter IMBALANCE_TOL[0] = 1.100000
ZOLTAN Parameter AUTO_MIGRATE = FALSE
ZOLTAN Parameter MIGRATE_ONLY_PROC_CHANGES = 1
ZOLTAN Parameter OBJ_WEIGHT_DIM = 1
ZOLTAN Parameter EDGE_WEIGHT_DIM = 0
ZOLTAN Parameter DEBUG_LEVEL = 1
ZOLTAN Parameter DEBUG_PROCESSOR = 0
ZOLTAN Parameter DETERMINISTIC = TRUE
ZOLTAN Parameter TIMER = 1 (wall)
ZOLTAN Parameter NUM_GID_ENTRIES = 1
ZOLTAN Parameter NUM_LID_ENTRIES = 0
ZOLTAN Parameter RETURN_LISTS = IMPORT AND EXPORT
ZOLTAN Parameter NUM_GLOBAL_PARTS = 179
ZOLTAN Parameter NUM_LOCAL_PARTS = -1
ZOLTAN Parameter REMAP = 1
ZOLTAN Parameter SEED = 1423941669 (1423941669)
ZOLTAN Parameter LB_APPROACH = repartition
ZOLTAN Parameter RCB_OVERALLOC = 1.200000
ZOLTAN Parameter RCB_REUSE = 0
ZOLTAN Parameter CHECK_GEOM = 1
ZOLTAN Parameter RCB_OUTPUT_LEVEL = 0
ZOLTAN Parameter KEEP_CUTS = 0
ZOLTAN Parameter RCB_LOCK_DIRECTIONS = 0
ZOLTAN Parameter RCB_SET_DIRECTIONS = 0
ZOLTAN Parameter RCB_RECTILINEAR_BLOCKS = 0
ZOLTAN Parameter OBJ_WEIGHTS_COMPARABLE = 0
ZOLTAN Parameter RCB_MULTICRITERIA_NORM = 1
ZOLTAN Parameter RCB_MAX_ASPECT_RATIO = 10.000000
ZOLTAN Parameter AVERAGE_CUTS = 0
ZOLTAN Parameter RANDOM_PIVOTS = 0
ZOLTAN Parameter RCB_RECOMPUTE_BOX = 0
ZOLTAN Parameter REDUCE_DIMENSIONS = 0
ZOLTAN Parameter DEGENERATE_RATIO = 0.000000
ZOLTAN Parameter FINAL_OUTPUT = 0
ZOLTAN Parameter KEEP_CUTS = 0
ZOLTAN Parameter REDUCE_DIMENSIONS = 0
ZOLTAN Parameter DEGENERATE_RATIO = 10.000000
Zoltan (level 4) : time required = 5.726814e-02
```

```
ML_Gen_MultiLevelHierarchy (level 4) : Gen Restriction and Prolongator
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
ML_Aggregate_Coarsen (level 4) begins
ML_Aggregate_CoarsenUncoupled : current level = 4
ML_Aggregate_CoarsenUncoupled : current eps = 0.000000e+00
Aggregation(UVB) : Total nonzeros = 4584792 (Nrows=46025)
Aggregation(UVB) : Amalgamated matrix done
Aggregation(UC) : Phase 0 - no. of bdry pts = 0
Aggregation(UC) : Phase 1 - nodes aggregated = 5416 (9205)
Aggregation(UC) : Phase 1 - total aggregates = 767
Aggregation(UC_Phase2_3) : Phase 1 - nodes aggregated = 5416
Aggregation(UC_Phase2_3) : Phase 1 - total aggregates = 767
Aggregation(UC_Phase2_3) : Phase 2a- additional aggregates = 168
Aggregation(UC_Phase2_3) : Phase 2 - total aggregates = 935
Aggregation(UC_Phase2_3) : Phase 2 - boundary nodes = 0
Aggregation(UC_Phase2_3) : Phase 3 - leftovers = 0 and singletons = 0

Prolongator/Restriction smoother (level 4) : damping = 0.000e+00 , sweeps = 1
Gen_Prolongator (level 4) : not smoothing prolongator
ML_Gen_MultilevelHierarchy: Projecting node coordinates from level 4 to level 5
Repartitioning (level 5): min rows per proc = 256
Repartitioning (level 5): largest max/min ratio = 1.200e+00
Repartitioning (level 5): max #rows (global) that fits on one proc = 5000
Repartitioning (level 5): #proc to use in repartitioning = 1
Smoothed Aggregation : operator complexity = 2.221882e+00.
Time to build the hierarchy = 8.50664 (s)
Number of actual levels : 6

Smoother (level 0) : # global rows = 1343488000, # estim. global nnz = 181252915200
Smoother (level 0) : IFPACK, type='ILU',
Smoother (level 0) : both,overlap=0
Smoother (level 0) : level-of-fill=0,rel. threshold=1,abs. threshold=0
Smoother (level 0) : Setup time : 0.999883 (s)

Smoother (level 1) : # global rows = 70043520, # estim. global nnz = 9061678016
Smoother (level 1) : IFPACK, type='ILU',
Smoother (level 1) : both,overlap=0
Smoother (level 1) : level-of-fill=0,rel. threshold=1,abs. threshold=0
Smoother (level 1) : Setup time : 0.198701 (s)

Smoother (level 2) : # global rows = 4545920, # estim. global nnz = 531425000
Smoother (level 2) : IFPACK, type='ILU',
Smoother (level 2) : both,overlap=1
Smoother (level 2) : level-of-fill=0,rel. threshold=1,abs. threshold=0
Smoother (level 2) : Setup time : 0.395695 (s)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
Smoother (level 3) : # global rows = 462075, # estim. global nnz = 47854528
Smoother (level 3) : IFPACK, type='ILU',
Smoother (level 3) : both,overlap=1
Smoother (level 3) : level-of-fill=0,rel. threshold=1,abs. threshold=0
Smoother (level 3) : Setup time : 0.182468 (s)

Smoother (level 4) : # global rows = 46025, # estim. global nnz = 4584792
Smoother (level 4) : IFPACK, type='ILU',
Smoother (level 4) : both,overlap=1
Smoother (level 4) : level-of-fill=0,rel. threshold=1,abs. threshold=0
Smoother (level 4) : Setup time : 0.153977 (s)

Amesos (level 5) : NumGlobalRows = 4675
Amesos (level 5) : NumGlobalNonzeros = 451128
Amesos (level 5) : Fill-in = 2.06413 %
Amesos (level 5) : Building KLU
Amesos (level 5) : Time for factorization = 2.67305 (s)

WARNING: Parameter "energy minimization: type" 2 [unused] is unused
WARNING: Parameter "smoother: list (level 0)" [unused] is unused
WARNING: Parameter "smoother: list (level 1)" [unused] is unused
WARNING: Parameter "coarse: list" [unused] is unused
WARNING: Parameter "aggregation: list (level 0)" [unused] is unused
WARNING: Parameter "aggregation: list (level 1)" [unused] is unused
WARNING: Parameter "aggregation: list (level 2)" [unused] is unused
WARNING: Parameter "aggregation: list (level 3)" [unused] is unused
WARNING: Parameter "aggregation: list (level 4)" [unused] is unused
WARNING: Parameter "aggregation: list (level 5)" [unused] is unused
WARNING: Parameter "aggregation: list (level 6)" [unused] is unused
WARNING: Parameter "aggregation: list (level 7)" [unused] is unused
WARNING: Parameter "aggregation: list (level 8)" [unused] is unused
WARNING: Parameter "aggregation: list (level 9)" [unused] is unused
WARNING: Parameter "smoother: list (level 2)" [unused] is unused
WARNING: Parameter "smoother: list (level 3)" [unused] is unused
WARNING: Parameter "smoother: list (level 4)" [unused] is unused
```

Cumulative timing for construction so far:

- for initial setup = 451.649 (s)
- for hierarchy setup = 9237.18 (s)
- for smoothers setup = 828.621 (s)
- for coarse setup = 698.491 (s)
- for final setup = 52.5722 (s)

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
*****
***** Problem: Epetra::CrsMatrix
***** Preconditioned GMRES solution
***** Teko::PreconditionerLinearOp<double>
***** No scaling
*****  
  
iter:      0      residual = 1.000000e+00
iter:     10      residual = 2.604812e-01
iter:     20      residual = 4.042929e-02
iter:     30      residual = 1.012947e-02
iter:     40      residual = 4.599561e-03
iter:     50      residual = 1.728580e-03
iter:     58      residual = 9.290841e-04  
  
Solution time: 6.124757 (sec.)
total iterations: 58  
  
Field Average      Maximum (@IP)      Minimum (@IP)
UX -7.72078643e-08 2.23316890e+01 -2.23317117e+01
UY -5.08567607e-08 2.23317097e+01 -2.23316906e+01
UZ 1.02251330e+01 3.11821340e+01 1.66023420e+00
*****  
-- Status Test Results --
**.....OR Combination ->
**.....AND Combination ->
Converged....F-Norm = 4.190e-09 < 1.000e-03
          (Unscaled Two-Norm, Absolute Tolerance)
**.....WRMS-Norm = 6.681e+00 < 1
          (Min Step Size: 1.000e+00 >= 1)
**.....Number of Iterations = 3 < 10
**.....Finite Number Check (Two-Norm F) = Finite
*****  
  
-- Nonlinear Solver Step 3 --
||F|| = 4.190e-09 step = 1.000e+00 dx = 7.708e+01
*****
```

Note that a large fraction of time is spent in the hierarchy construction and rebalancing algorithm in the v-cycle of the algebraic multilevel preconditioner. This will be a focus in the Q3/Q4 improvements.

0.1.7 Developments, Enhancements, and Q4 Benchmark Results

0.1.8 Q4 Baseline Problem Results

0.1.9 Application Summary of Results and Analysis

0.2 VASP - Materials Project

0.2.1 Introduction

VASP [?, ?, ?] is a sophisticated package for performing atomic scale material modeling using density functional theory (DFT). It has very broad applicability to, e.g., solid-state materials, molecules and nanostructures, and can treat both semiconductors and metals. VASP uses a plane wave representation of the wavefunctions and either pseudopotentials or the projector-augmented wave method. Both zero temperature static and finite temperature molecular dynamics based methods are implemented as well as a large variety of analysis methods and post-processing. VASP is one of the most popular DFT codes owing to its implementation of modern methods, efficiency, high quality pseudopotential library, and a very good set of defaults. This combination allows DFT results for new systems to be obtained very rapidly and without some of the complexity displayed by traditional plane wave DFT codes.

VASP is written using FORTRAN90 and can be run either serially or in parallel using message passing based on MPI. The code makes very extensive use of linear algebra via the BLAS and LAPACK numerical libraries. Fourier transforms are provided via the FFTW package. Custom code is used for distributed Fourier transforms, to exploit the spherical cutoff in the plane wave coefficients in reciprocal (Fourier) space.

The main author of the code is Prof. Dr. Georg Kresse of the University of Vienna, Austria. An extensive description, list of other contributors, and manual is available online (www.vasp.at). The VASP source code is licensed.

0.2.2 Background and Motivation

0.2.3 Capability Overview

Time to solution by using more processors.

0.2.4 Science Driver for Metric Problem

VASP is the first code to be examined by the Materials Project for a large scale survey of pseudo-potentials of inorganic crystalline compounds. The project is a high-throughput computing effort that involves hundreds of thousands of independent physical systems. For each compound a set of base thermal, mechanical, and spectroscopic properties are calculated and the metric formulated here involves a small number of the runs that make up the overall workflow. The systems of interest for the VASP calculations are chosen from the same source as that of , which is the ICSD (Inorganic Crystal Structures Database). In order to cover a representative span of these inputs a variety of band-gaps and ionic-step convergence properties were chosen. Insulators, semiconductors, and conductors typically display distinct computational requirements in terms of the rate at which the calculation converges to a stable answer.

Due to the impracticality of performing a complete set of calculations for a given materials database, we have chosen four DFT prototypical calculations. These were provided by Kristin Persson of LBL and David Skinner of NERSC. Crucially these problems are run with exactly the same convergence criteria and have similar atom counts for the database calculations. Improvements in the performance of these calculations is therefore directly transferable to the whole database problem.

The numerical problem size is determined primarily by the atom count (electron count), the plane wave cutoff, and the number of k-points within the Brillouin zone. The test problems consist of relaxing the atoms, volume, and shape of the supercell, and finally determining the cohesive energy of the resultant fully relaxed supercell. Determining these properties takes the majority of the computational time in a typical materials database problem. In a research problem these results would be utilized to, e.g. construct a phase diagram for the material. The electronic structure is determined using VASP’s “FAST” algorithm, which uses a combination of Davidson and the residual-minimization methods. The calculations are performed spin-polarized.

0.2.5 Software Description: Model and Algorithm

Needs to be added in Q3.

Problem	Machine	No. of cores	Total FP_INS	Total L2_DCM	Real (s)	User (s)
engine-8296	Hopper	24	3.327e13	1.969e11	3126.1	3125.7
		48	4.031e13	2.440e11	2218.8	2218.4
		72	3.273e13	2.080e11	1315.3	1315.1
		96	3.355e13	2.254e11	1096.8	1096.6
engine-11691	Hopper	24	1.954e13	1.269e11	2189.5	2189.2
		48	2.000e13	1.332e11	1260.3	1260.1
		72	2.057e13	1.490e11	947.4	947.2
		96	2.129e13	1.542e11	806.1	805.4
engine-12494	Hopper	24	1.140e13	7.771e10	1107.6	1107.5
		48	1.164e13	8.236e10	669.2	669.0
engine-14636	Hopper	24	4.594e13	3.113e11	3678.1	3677.8
		48	4.647e13	3.231e11	2192.5	2192.3
		72	4.741e13	3.390e11	1713.9	1713.8
		96	4.856e13	3.570e11	1338.2	1337.9

Table 7: Problem performance results for Hopper at NERSC. Measurements for PAPI hardware counters are given for the entire simulation including all startup, initialization, and I/O costs. The engine-12494 problem on 72 and 96 cores did not run with our chosen values of NPAR=3 & 4 , respectively, due limitations of the simulation code.

0.2.6 Q2 Baseline Problem Results

The inputs for the baseline problems are provided in the Appendix of this document. In Q3 we will describe an algorithm to parallelize the current computational algorithm over the k-points. Here we present the measured machine events for the Q2 baseline runs.

Problem	Machine	No. of cores	Total FP_INS	Total L2_DCM	Real (s)	User (s)
engine-8296	Jaguar	16	6.314e13	3.869e11	5632.2	5628.9
		32	5.922e13	3.816e11	2784.1	2783.9
		48	7.435e13	4.840e11	4679.9	4669.0
		64	5.187e13	3.556e11	1395.1	1394.6
		80	7.079e13	4.894e11	1521.3	1518.6
		96	6.617e13	4.420e11	1234.2	1229.5
engine-11691	Jaguar	16	3.758e13	2.019e11	2804.7	2804.5
		32	3.818e13	2.119e11	1617.5	1616.5
		48	3.874e13	2.168e11	1798.9	1798.4
		64	4.000e13	2.306e11	937.4	936.8
		80	4.111e13	2.334e11	806.1	802.3
		96	4.125e13	2.366e11	705.0	702.9
engine-12494	Jaguar	16	3.757e13	2.028e11	2994.2	2994.1
		32	3.811e13	2.122e11	1619.5	1619.3
		48	3.878e13	2.169e11	1796.5	1796.0
		64	4.000e13	2.311e11	909.9	909.1
		80	4.104e13	2.329e11	810.4	807.0
		96	4.123e13	2.364e11	695.4	693.4
engine-14636	Jaguar	16	8.952e13	5.957e11	6209.6	6209.6
		32	9.067e13	6.066e11	3313.6	3312.9
		48	9.208e13	6.141e11	3073.9	3087.0
		64	9.386e13	6.559e11	1878.9	1878.1
		80	9.347e13	6.388e11	1565.0	1562.3
		96	9.541e13	6.374e11	1389.0	1384.2

Table 8: Problem performance results for Jaguar at OLCF. Measurements for PAPI hardware counters are given for the entire simulation including all startup, initialization, and I/O costs.

0.3 NIMROD

0.3.1 Introduction

Nuclear fusion, the energy that powers the stars and our sun, is one of the few sources of energy that is plentiful for the long term and has the potential for being environmentally benign. Conditions that lead to sufficient fusion reactions require temperatures approximately 1 millions times hotter, and pressures somewhat greater, than standard atmospheric conditions. In such extreme conditions, material is in the plasma state, and the charged particles that compose plasma are deflected by the Lorentz force when they move perpendicular to magnetic field-lines. This simple physical concept is the basis for magnetic confinement, where tailored magnetic configurations serve to contain and thermally insulate the hot plasma particles. Scientific and engineering research over the past six decades puts magnetic fusion energy (MFE) on the threshold of producing self-heated conditions, and this is the primary objective of the international ITER experiment that is under construction in France [?].

Like the highest-performance magnetic confinement experiments in existence today, the ITER experiment is a tokamak configuration. Tokamaks are characterized by a large toroidal component of magnetic field (**B**) and by toroidal electrical current running in the plasma to provide a smaller poloidal component of **B**. ('Toroidal' refers to the direction running around the geometric axis, and 'poloidal' refers to the plane normal to the toroidal direction.) The high pressure achieved in present-day tokamak plasmas largely results from the transport barrier that forms just inside the separatrix of the equilibrium poloidal components of **B**. Magnetic field-lines are contained inside the toroidally shaped separatrix surface, and field-lines outside this surface eventually intercept the chamber wall. Plasma pressure rises sharply just inside the separatrix, and that effect is beneficial for achieving fusion-relevant conditions. However, free energy associated with large pressure and current gradients in this region is prone to excite edge-localized modes (ELMs), a perturbation that breaks toroidal symmetry to release some of the stored internal energy in discrete bursts. While the ELMs may be viewed as a self-regulating mechanism of the plasma, the projected amount of released energy for certain classes of ELM events in a device the size of ITER raises concern for the longevity of the plasma-facing components of the chamber. In fact, the steady power loading in post-ITER demonstration reactors is expected to reach the very limit of any known material, and the intermittent behavior of ELMs would exceed the limit. As such, ELMs are identified in the 2002 Fusion Summer Study [?] as one of the critical magnetohydrodynamics (MHD) theory topics for assessing burning plasma designs, and they remain an area of central importance for fusion theory. ELMs and predictive simulations of them are noted as a research priority in the fusion community's report from the Research Needs for Magnetic Fusion Energy Science workshop held in June, 2009 [?].

0.3.2 Background and Motivation

There have been significant advances in our understanding of many properties of ELM events. That the bursts result from magnetohydrodynamic instability has been well established through specialized linear analysis [?, ?] and detailed validation efforts [?]. An early nonlinear study solves compressible reduced-MHD equations in a global geometry with the core-plasma region removed [?]. The equilibrium generated in the computation excites ballooning instabilities at low toroidal mode index (n) that drive poloidal flows into the divertor late in the nonlinear stage. Another study applies a reduced two-fluid model to selected toroidal harmonics in a narrow region around the separatrix [?] and finds a toroidally localized structure emerging late in the evolution. The localization represents nonlinear coupling of multiple unstable modes and is consistent (in the context of the selected harmonics) with laboratory observations of filamentary structures. Full-geometry computations with a non-reduced two-fluid model and initial conditions from profiles fit to actual experiments indicate a similar localization process, but computational limitations at the time precluded analysis of energy loss [?]. Since the full-geometry two-fluid computations were first run, there has been simulation progress on a number of nonlinear topics. Reduced two-fluid modeling in a domain that has a limited radial extent considers the influence of magnetic reconnection on the amount of energy lost during an event [?]. Full-geometry computations with MHD and with a two-fluid drift model investigate topological changes in the magnetic field as fully developed ELM events interact with the separatrix of the equilibrium field [?]. Other ELM-relevant MHD studies support nonlinear analytical predictions of ballooning behavior [?].

0.3.3 Science Driver for the Metric Problem

The computations used for the metric problem considered here are based on a recent MHD study of the effects of edge current density [?]. The localized parallel current that arises inside the separatrix is a source of free energy and contributes the ‘peeling’ part of the underlying MHD instabilities. However, the results in Ref. [?] show that when this current is sufficiently large that the shear in the magnetic field reverses, MHD growth rates are reduced. A nonlinear computation described in this article shows that the smaller growth rate persists through the intermediate nonlinear phase. It is important to know whether the slower evolution leads to a smaller ELM burst in terms of lost energy, but the amount of computational time per unit physical time increases with perturbation amplitude, and the simulation run with the NIMROD code [?] becomes too costly to fully answer this question.

In the early phase of this numerical simulation, amplitudes of the perturbations are small, and there is little coupling among the different Fourier harmonics over the toroidal angle. During this

phase, the computation progresses much like a collection of independent linear computations. As the perturbation amplitude increases, the energy exchanged through nonlinear coupling increases, and the perturbations begin to alter the equilibrium profiles that drive the instability. With the NIMROD algorithm, described in the following section, the algebraic systems solved at each time-step are linear systems. However, as nonlinear effects become important physically, matrices that result from the implicit advances need to be recomputed and refactored frequently. The profile changes rapidly in the strongly nonlinear phase, and when the matrices need to be recomputed more than once per every ten steps (approximately), the code spends at least as much time in matrix computation and factoring as it does in solving the systems.

The three runs developed for the metric problem have been selected to represent the primary computational improvements that are needed for the edge-current study: improved resolution and greater efficiency in strongly nonlinear conditions. The 'medium linear' problem has a 40x65 mesh of biquintic finite elements to represent the spatial variations of 8 physical quantities (three vector-components each for magnetic field and flow velocity, plus the two scalar fields of number density and temperature) for a single Fourier component. That amounts to approximately 500,000 complex degrees of freedom. NIMROD's advance allows solving each physical field separately, hence smaller algebraic systems. Moreover, the matrices are computed and factored only once in a linear computation. The 'large linear' problem has better poloidal resolution with an 80x128 mesh of biquintic elements. Comparison of the two linear computations (and extending beyond the 'large' problem) provides information on weak scaling. Improvements in poloidal resolution are needed to run conditions with more realistic dimensionless dissipation parameters. The third version of the metric problem has the poloidal resolution of the 'medium linear' problem and includes 11 coupled Fourier harmonics, $0 \leq n \leq 10$. The toroidal resolution of this test is modest relative to the $0 \leq n \leq 42$ components used in Ref. [?], but it is sufficient to illustrate the computational challenges associated with implicit nonlinear ELM computations.

0.3.4 Software Description: Model and Algorithm

The NIMROD code solves fluid-based models for macroscopic dynamics in magnetized plasma. Several applications rely on our two-fluid modeling capability [?] or our energetic-particle model [?], but the critical computational challenges are evident from resistive-MHD with temperature-dependent electrical resistivity and anisotropic thermal conduction, as used in Ref. [?]. This system of equations includes a continuity equation for particle number density (n), an evolution equation for plasma flow-velocity (\mathbf{v}), and a temperature-evolution equation:

$$\frac{\partial n}{\partial t} = -\nabla \cdot (n\mathbf{v}) ,$$

$$m_i n \left(\frac{\partial}{\partial t} \mathbf{v} + \mathbf{v} \cdot \nabla \mathbf{v} \right) = \mathbf{J} \times \mathbf{B} - 2\nabla(nT) + \nabla \cdot \mathbf{v} m_i n \nabla \mathbf{v}, \text{ and}$$

$$\frac{3}{2} n \left(\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T \right) = -nT \nabla \cdot \mathbf{v} + \nabla \cdot [(\chi - \chi_{iso}) \mathbf{b} \mathbf{b} + \chi_{iso} \mathbf{I}] \cdot n \nabla T,$$

where \mathbf{b} is the temporally evolving unit vector-field in the direction of the magnetic field. The coefficient for parallel thermal diffusivity (χ) is orders of magnitude larger than a representative value for the isotropic component(χ_{iso}) in high-temperature plasma. This property of thermal transport and the perpendicular nature of the Lorentz force effect extremely anisotropic conditions in magnetized plasma. The evolution of the magnetic field is governed by Faraday's law with the resistive-MHD Ohm's law relation for electric field,

$$\frac{\partial}{\partial t} \mathbf{B} = \nabla \times (\mathbf{V} \times \mathbf{B} - \eta \mathbf{J}),$$

where η is the electrical resistivity, and the current-density \mathbf{J} follows from Ampere's law at low frequency, $\mathbf{J} = \nabla \times \mathbf{B}$.

The NIMROD code solves this system of equations for a specified set of boundary and initial conditions using a semi-implicit method, where the flow velocity is staggered by half a time-step from the other physical fields. 'Semi-implicit' refers to adding a spatial differential operator to the partial derivative of \mathbf{v} with respect to time. This term is multiplied by Δt^2 , so the system remains numerically consistent. As described in Ref. [?], and references therein, the approach effectively makes inertia dependent on the wavenumber of a perturbation and slows small-wavelength waves so that the static-background linear ideal-MHD system does not have a CFL stability restriction. This is important for modeling magnetically confined plasma, where MHD waves propagate globally on much shorter time-scales than the nonlinear evolution of interest. Its advantage over fully implicit methods is that it does not require all fields to be solved simultaneously, which lessens the burden on numerical linear algebra software. Nonetheless, the global coupling represented in the velocity advance and in the advance of other fields leads to ill-conditioned matrices. Owing to the toroidal symmetry of the geometry and the background fields, toroidal Fourier components are independent in linear computations. The sparse, ill-conditioned matrices that represent implicit coupling over the poloidal plane for a given Fourier component are solved in parallel with the SuperLU_DIST software library [?]. Nonlinear evolution breaks the toroidal symmetry, and NIMROD uses "matrix-free" iterative Krylov-space methods (CG or GMRES) to solve fully coupled systems. Here, SuperLU_DIST is applied to blocks of the full algebraic system as a preconditioning step, and like the linear computations, each block represents the coupling over the poloidal plane for each Fourier component.

The spatial representation uses spectral finite elements [?] for the poloidal plane and finite

Fourier series for the periodic toroidal direction. The spectral methods allow convergence on the extremely anisotropic conditions of magnetized high-temperature plasma and allow very accurate representation of the magnetic divergence constraint from Maxwell's equations [?]. The Fourier representation in the toroidal angle implies that coupling in nonlinear computations leads to dense blocks from convolutions. However, the coefficients are proportional to the size of the nonlinear perturbation relative to the size of the background, so the coupling tends to be weaker than the coupling over the poloidal plane. Thus, matrix-free Krylov iteration is a suitable approach, provided that the strong coupling over the poloidal plane is relaxed through preconditioning. The burden for the external direct solve is reduced by 'static condensation,' i.e. partitioning and eliminating the degrees of freedom from element interiors within NIMROD, itself. The terms that couple different Fourier components in nonlinear applications are computed on a uniform mesh over the toroidal angle, and fast Fourier transforms (FFTs) are used to relate Fourier coefficients and values on the toroidal mesh. The transformations are arranged to provide dealiasing for quadratic terms.

Parallel computation in the NIMROD code follows the distributed-memory approach with spatially 3D domain decomposition, and the Message Passing Interface (MPI) library is used for all parallel communication. The Fourier components are collected in 'processor layers,' and collective MPI_ALLTOALLV communication is used with appropriate communicator groups so that the independent FFT and toroidal-mesh computations proceed simultaneously with serial operations. The poloidal mesh is broken into 'grid blocks' with the same decomposition for all processor layers. Parallel communication for rhs computations in NIMROD uses asynchronous point-to-point communication within separate communicator groups for each processor-layer. The use of SuperLU_DIST in linear computations and as a preconditioner in nonlinear computations is such that it works on algebraic systems for one Fourier component at a time, and different processor-layer groups in nonlinear computations invoke SuperLU_DIST simultaneously.

0.3.5 Q2 Baseline Problem Results

The Q2 baseline results for NIMROD ELM computations have been run on the Cray XE6 (Hopper) with its 24 cores per node at the National Energy Research Scientific Computing Center (NERSC) at Lawrence-Berkeley National Laboratory. The NIMROD executable was created with the Portland Group compiler using the settings and modules shown the appendices. The KRP routines have been used to monitor distinct phases of the computations: 1) matrix-element computation in NIMROD, 2) static condensation operations combining factoring and solution of small dense matrices, 3) computation of the 'right-hand-side' vector for advancing the solution field, 4) parallel LU factorization performed by SuperLU_DIST, and 5) parallel LU solution performed by SuperLU_DIST. Evolution between these distinct phases is also timed and summed with the distinct phases to ag-

gregate information for the computation, apart from initialization. Other computations without the distinct-phase monitoring confirm that the amount of overhead from the distinct-phase counters is negligible in the aggregate data. Note that the 'rhs' monitor in nonlinear computations includes the computation of Krylov direction vectors during matrix-free iteration, FFT computations, and the collective communication associated with the toroidal-mesh computations.

Table 9 shows the KRP-monitor results for the medium linear computation, which was run on a single node using all 24 cores of the node. The 40x65 mesh of biquintic elements is decomposed into a 4x6 mesh of grid blocks for parallelization. The 'fe matrix' and 'LU fact' operations are performed only once during the initialization for this 100-step computation.

Table 9: Hardware-counter for the 24-core medium linear computation.

	INS	FP_INS	L2_DCM	microseconds
fe matrix	1.77×10^{11}	8.23×10^{10}	1.73×10^9	3,486,972
fe rhs	1.43×10^{11}	2.52×10^{10}	1.60×10^9	3,290,672
st cond	1.12×10^{11}	3.92×10^{10}	2.19×10^8	2,087,658
LU fact	7.90×10^{11}	8.62×10^{10}	5.30×10^8	10,906,015
LU solve	9.66×10^{11}	5.05×10^{10}	1.52×10^9	16,719,071
total	3.24×10^{12}	4.02×10^{11}	6.57×10^9	55,025,012

Table 10 shows the KRP-monitor results for the large linear computation, which was run on 128 cores using 16 cores per node. The 80x128 mesh of biquintic elements is decomposed into an 8x16 mesh of grid blocks for parallelization.

Table 10: Hardware-counter for the 128-core large linear computation.

	INS	FP_INS	L2_DCM	microseconds
fe matrix	6.99×10^{11}	3.24×10^{11}	6.46×10^9	2,785,066
fe rhs	5.66×10^{11}	9.91×10^{10}	4.58×10^9	2,580,768
st cond	4.40×10^{11}	1.55×10^{11}	8.61×10^8	2,073,935
LU fact	1.61×10^{13}	5.19×10^{11}	9.14×10^9	39,878,358
LU solve	1.50×10^{13}	2.17×10^{11}	8.28×10^9	33,679,073
total	4.24×10^{13}	1.78×10^{12}	3.79×10^{10}	111,212,611

The small nonlinear computation has the same 40x65 mesh as the medium linear computation and the same 4x6 grid-block decomposition. The 11 Fourier components are distributed on 6 processor layers. The computation ran on 6 nodes using all 24 cores per node for 144 total cores. The data for the first segment of the computation, 709 steps, is presented in Table 11. Here,

the interaction among Fourier components is weak, and matrices are recomputed and refactored infrequently. The data for the last 83 steps of the computation, which took as much wall-clock time as the first 709, is presented in Table 12. By this point, the computation has entered the more strongly nonlinear phase, and a much greater fraction of the computational time is spent computing and refactoring matrices. Efforts to improve performance on these operations will have the greatest benefit for nonlinear applications.

Table 11: Hardware-counter for first segment of the nonlinear computation.

	INS	FP_INS	L2_DCM	microseconds
fe matrix	6.54×10^{12}	3.50×10^{12}	5.57×10^{10}	18,235,885
fe rhs	2.58×10^{14}	5.09×10^{13}	5.06×10^{11}	554,521,136
st cond	9.03×10^{12}	3.80×10^{12}	8.83×10^9	26,847,392
LU fact	1.70×10^{13}	2.92×10^{12}	9.87×10^9	36,290,568
LU solve	1.93×10^{14}	1.27×10^{13}	2.26×10^{11}	493,078,556
total	7.17×10^{14}	1.01×10^{14}	8.85×10^{11}	1,649,003,458

Table 12: Hardware-counter for last segment of the nonlinear computation.

	INS	FP_INS	L2_DCM	microseconds
fe matrix	1.70×10^{14}	9.02×10^{13}	1.41×10^{12}	474,487,899
fe rhs	1.14×10^{14}	6.78×10^{12}	6.47×10^{10}	82,546,997
st cond	3.29×10^{13}	1.32×10^{13}	8.80×10^{10}	126,397,723
LU fact	3.73×10^{14}	8.99×10^{13}	1.94×10^{11}	701,507,455
LU solve	3.03×10^{13}	2.07×10^{12}	3.35×10^{10}	76,223,937
total	7.80×10^{14}	2.06×10^{14}	1.82×10^{12}	1,663,796,057

0.3.6 Science Driver for Q2 Baseline Problem Two

Wave-particle interactions are critical to understanding plasma instabilities in high temperature plasmas, in both terrestrial devices such as tokamaks and in space, for example magnetotails and solar events. The presence of sufficiently energetic particles can sometimes stabilize instabilities beyond expected ideal MHD stability boundaries leading to a build up of stored energy until a secondary stability threshold is reached, at which point a more violent disruption occurs such as in tokamak sawtooth events⁴. This energy build up and crash are not well understood due, in

⁴M. Choi, et al., “Sawtooth control using beam ions accelerated by fast waves....”, Physics of Plasmas **14**, 112517 (2007)

part, to the complex dynamics of energetic particles and their sensitivity to details of geometry and distribution function. Alternatively, the presence of energetic particles may excite energetic particle driven modes such as fishbone modes and toroidal Alfvén eigenmodes (TAE). These energetic particle modes such as TAE's are also sensitive to the details of geometry and distribution function. This dependence on details of geometry and the kinetic nature of the instabilities limits the applicability of analytic theories to address the observations made in experiments. Initial value simulations using the realistic geometry of experiments can bridge this gap between analytic theory and experimental observations.

Hybrid kinetic-magnetohydrodynamic (MHD) simulations⁵ using the NIMROD code⁶ can address this need for a variety of plasma devices from tokamaks to Field Reverse Configurations (FRC). The hybrid kinetic-MHD model modifies the MHD momentum equation by the addition of an energetic particle pressure tensor \mathbf{p}_h

$$(31) \quad \rho \left(\frac{\partial U}{\partial t} + U \cdot \nabla U \right) = J \times B - \nabla p_b - \nabla \cdot p_h$$

where p_b is the bulk fluid pressure and \mathbf{p}_h is the energetic particle pressure tensor. In this modified momentum equation, the mass density ρ and the flow U include the energetic particles as well as the bulk ions and electrons. The energetic species is represented by δf PIC particles and integrated along drift kinetic equations of motion

$$(32) \quad \dot{\mathbf{x}} = v_{\parallel} \hat{b} + v_D + v_{E \times B}$$

$$(33) \quad v_D = \frac{m}{eB^4} \left(2 + \frac{v_{\perp}^2}{2} \right) \left(\mathbf{B} \times \nabla \frac{B^2}{2} \right) + \frac{\mu_0 m^2}{eB^2} J_{\perp}$$

$$(34) \quad v_{E \times B} = \frac{E \times B}{B^2}$$

$$(35) \quad m \dot{v}_{\parallel} = -\hat{b} \cdot (\mu \nabla B - e E).$$

A CGL-like pressure tensor is computed by taking δf pressure moments

$$(36) \quad \delta p_h = \begin{pmatrix} \delta p_{\perp} & 0 & 0 \\ 0 & \delta p_{\perp} & 0 \\ 0 & 0 & \delta p_{\parallel} \end{pmatrix}$$

⁵C. C. Kim, “Impact of velocity space distribution on hybrid kinetic-mhd simulations....”, Physics of Plasmas **15**, 072507 (2008)

⁶C.R. Sovinec, et al., “Nonlinear Magnetohydrodynamics with High-order Finite Elements”, Journal of Computational Phycis **195**, 355 (2004)

where $\delta p_{\perp} = \int \mu B \delta f d^3 v$ and $\delta p_{\parallel} = \int m_h v^2 \parallel \delta f d^3 v$.

The initial metric problem is taken from a parametric study of the stability space of the ‘hybrid discharge’ in the DIII-D tokamak⁷. This case uses an equilibrium reconstruction from shot #125476 from the DIII-D database to examine the effect of energetic particles on the stability of high performance ‘hybrid discharge’ operating scenario that has been proposed for ITER, an international tokamak experiment that is currently under construction. This parameter study reveals that the presence of a modest non-maxwellian energetic particle population may be responsible for the sensitivity to β_N , a normalized measure of energy density in the tokamak, placing a significant limit on the energy density that is achievable.

0.3.7 Q2 Baseline Result -Problem Two

This simple weak scaling plots the time in microseconds taken by the particle advance portion of the hybrid kinetic-MHD simulation over 50 time steps for 16,32,64,128,256 processors as the particle number is increased from 2M,4M,8M,16M,32M . These cases were run on Jaguar at the ORNL OLCF. There is a clear performance break beyond 16 processors as internode communication must be performed. Beyond 16 processors, the code shows reasonable linear scaling for the modest number of processors.

0.3.8 Science Driver for Q2 Baseline Problem Three

Current drive methods based on magnetic helicity injection to form and sustain magnetically confined plasmas have been an important area of study for compact tori. Compact tori have minimal or no central column needed to drive the plasma via a transformer and therefore, rely on alternate plasma-drive techniques like magnetic helicity injection. Helicity injection current drive exploits the plasma’s tendency to self-organize itself toward a state of minimum potential energy via magnetohydrodynamic (MHD) activity subject to the constraint of magnetic helicity conservation, which is a measure of the (self) linkage or twistedness of magnetic flux [?, ?, ?]. The resulting minimum energy state is one in which all the plasma currents are aligned with the magnetic field giving rise to a globally uniform current profile. Although the final ‘relaxed’ state attained by the magnetic configuration is well understood, it is not clear how the plasma dynamically evolves toward this quiescent state. Therefore, understanding the detailed dynamics of magnetic helicity injection

⁷D. P. Brennan, C. C. Kim, and R. J. LaHay, ”Energetic particle effects on n = 1 resistive MHD instabilities in a DIII-D hybrid discharge”, Nuclear Fusion **52**, 033004, (2012)

current drive poses an important milestone for magnetic confinement experiments alternate to the tokamak.

Helicity injected torus with steady induction (HIT-SI) [?, ?] is a recent installment in a series of helicity injection current-drive experiments [?, ?, ?]. HIT-SI injects magnetic helicity into the plasma confinement region, completely electrode-free by the use of two semi-toroidal helicity injectors attached on either end of the confinement vessel (figure 5). The injectors oscillate magnetic flux and electrical current in phase to generate a helical magnetic field that traverses the confinement region. The injected fields have odd toroidal parity, i.e., they flip polarity 180 degrees around the torus as flux comes out of one mouth of an injector and goes into the other mouth of the same injector. Driving the injectors 90 degrees out of phase achieves steady inductive helicity injection (SIHI). The electrodeless operation makes HIT-SI an ideal platform for clean studies of helicity injection current drive and magnetic relaxation. Since the device operates on non-axisymmetric current drive, all of the toroidal field and plasma current must be an outcome of magnetic relaxation.

HIT-SI has a limited number of probes and other diagnostic instruments. Additional diagnostics are costly, time-consuming to build and present a risk of introducing impurities into the plasma. Numerical simulations provide the best means to study the plasma phenomena. We propose to examine the detailed dynamics of toroidal plasma formation and sustainment in HIT-SI with 3D, zero-temperature MHD simulations. The MHD equations represent the interactions between the two-fluid plasma comprising electrons and ions with magnetic and electric fields generated externally to and internally by the plasma. The two-fluid MHD model captures the electron dynamics believed to play a key role in the relaxation phenomena as their increased mobility aides in the electrical current conduction along the magnetic field resulting in a more uniform current profile. The helicity injectors of the experiment are modeled as oscillating normal magnetic (B_n) and tangential electric (E_t) field boundary conditions (BC). Our calculations to date show that a large toroidally symmetric (axisymmetric) structure and significant plasma current are generated as a result of the applied BC simulating steady inductive helicity injection (SIHI) current drive.

0.3.9 Q2 Baseline Result -Problem Three

The present model assumes a uniform plasma density, zero plasma temperature and a fully ionized plasma. The plasma density is uniform in space and assumed to be static in time. Thus, only the plasma flow (v) and magnetic field (B) are time-advanced. Radiative effects and neutral physics are not accounted for and collisional effects only enter through the plasma resistivity and the fluid stress. In the presence of two-fluid (2fl) effects, the magnetic field is convected with the electron

fluid instead of the ion fluid and therefore, internal electric fields are generated due to the motion of the electrons. Electron inertial effects due to finite electron mass (m_e) can play a key role in the dynamics of magnetic relaxation and are included in the calculations. An enhanced ion to electron mass ratio, $m_i/m_e = 36$ is used instead of 3600. The enhanced electron mass makes the matrices more diagonally dominant and speeds up the computation considerably. A convergence study on m_i/m_e indicates the results to be insensitive for mass ratios as high as 100. Computation time increases linearly with m_i/m_e . Going to higher mass ratios requires scaling the problem up to tens of thousands of cores and understanding the scaling behavior.

2fl effects become significant when current sheets within which magnetic field topology changes, become comparable in size (width) to the ion inertial length d_i . For the physical parameters of interest, $d_i \lesssim 1$ cm. This type of resolution requires approximately 50 finite element (FE) cells in either direction in the poloidal plane (R, Z) and possibly as many as 88 Fourier modes to discretize the toroidal (azimuthal) direction. A high-order polynomial representation (pd) is usually required for the FE basis functions: $pd = 4$. For a high-resolution calculation like this, the total number of degrees of freedom (dof) can be as high as $(50 \times 4)^2 \times 88 \sim 3.5 \times 10^6$. Calculations performed on 48×48 poloidal meshes indicate good toroidal convergence in Fourier space for up to 22 modes. But, even with 22 Fourier modes, the toroidal direction is still under-resolved compared to the poloidal plane by a factor of three. For toroidal plasmas like HIT-SI's, the magnetic field lines make approximately the same number of toroidal and poloidal transits inside the confinement volume, resulting in a fieldline pitch of approximately one. Fully resolving the magnetic field pitch might give rise to interesting and unseen phenomena. Thus, once again the resolution requirements increase the problem size forcing us to understand scaling to a large number of processors and to learn where improvements can be made to increase the efficiency of the computation.

We also aim to investigate the plasma dynamics for ‘hotter’ plasmas and understand how the global trends scale as the temperature of the plasma is raised. In our calculations, this is achieved by decreasing the background electrical resistivity (η). Results to date indicate that both the plasma current and axisymmetric mode energy increase as the plasma gets more conductive. But, as this happens, the dynamics get more stiff; time scales in the problem further separate, current sheets become thinner and 2fl effects become more pronounced. Thus, the matter of both weak and strong scaling and computational efficiency re-emerges here as well.

To study the (strong) scaling behavior, we chose a single test problem for the Q2 baseline simulation. The baseline simulation uses a 48×48 mesh with bicubic polynomial representation ($pd = 3$) to discretize the poloidal plane and 11 Fourier modes for the toroidal direction, corresponding to 32 toroidal slices. Total dof = 228096. The physical parameters are equal background ion and electron densities of $n = 1.5 \times 10^{19} \text{ m}^{-3}$, magnetic fields of $B \leq 0.2 \text{ T}$ and flow speeds less than 100 km/sec. The diffusivities for the baseline are computed assuming an equal background

ion and electron temperatures, $T_i = T_e = 10$ eV yielding 25 and 260 m²/sec. for the resistive and viscous dissipation respectively. The baseline simulation is split into two groups: (1) with a parallel partition of $8 \times 8 \times 11 = 704$ and (2) with a parallel partition of $12 \times 12 \times 11 = 1584$. For each group, we perform strong scaling studies by assigning different parallel partitions to each processor. For group 1, the baseline is run thrice, with 176, 352 and 704 processors corresponding to 4, 2 and 1 parallel partitions per processor respectively. For group 2, we use 66, 198, 396, and 792 processors corresponding to 24, 8, 4 and 2 parallel partitions per processor respectively. For each case study, the system of equations is advanced for 100 time steps. This is not long enough to get to the interesting dynamic behavior, but still provides us with some insight about where, in the NIMROD code, the calculation spends the most of time. We collect timing statistics from the four core stages of NIMROD computations for the time-advance. These are (1) fluid advance, (2) get_rhs where the FE element integrations are carried out, (3) matrix_create where the operators (like the mass matrix) on the left hand side (LHS) of the equations are constructed, and (4) the remaining cpu time, which is mostly spent in the solver (not shown in the figure). Figure 6 shows results for two different performance metrics. For both figures, the vertical and horizontal axes represent the natural logarithm of the total number of cpu hours in μ sec and the natural logarithm of the number of processors (*nprocs*) used in each simulation. The black traces correspond to group 1 and the red traces to group 2. Both metrics indicate get_rhs calculations to be most computationally-intensive part of the simulations. Total cpu hours spent in constructing the LHS operators is approximately one tenth of get_rhs cpu hours. The initial stage of the computation, fluid_advance uses a negligible amount of cpu time compared to get_rhs and mat_create. The results of the test problem also show that get_rhs calculations is where parallel partitioning makes the most difference in terms of cpu time. This is expected since the computational load per processor should be linearly proportional the number of FE nodes, and thereby number of FE integrations contained in each parallel block. This can be seen by comparing the computation time of get_rhs between the middle point of the black trace and the last point of the red trace. These two data points have the same number of parallel partitions per processor and take the same amount of cpu time to compute their FE integrations. mat_create and fladv appear to scale with the total number of processors. The three components together constitute about 60-70% of the total computation time, the rest of which is spent in the solver, which is the part that does not scale well with increasing number of processors. In fact, results from group 1 indicate less than 5% of change in solver time as the parallel load per processor is reduced from 4 to 1. In addition, solver time does not necessarily decrease as the number of parallel partitions per processor is lowered. Thus, the solver is where we need to increase the computational efficiency if it is at all possible.

Our preliminary performance analyses indicate the finite element integrations and the solver to be the most time-consuming parts of NIMROD's computational machinery for the two-fluid MHD problem we are studying. While FE-component of NIMROD scales perfectly with the number of parallel partitions per processor, the solver is insensitive to the load per processor. Thus, the solver is where we have the greatest room for improvement. We estimate we can make gains up to a factor

two in the computation speed if we improve the computational efficiency of the FE-integrations by rearranging the code and the scaling behavior of the solver. With these improvements we anticipate completing the aforementioned scans in less than 10^6 Jaguar wall clocks hours.

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

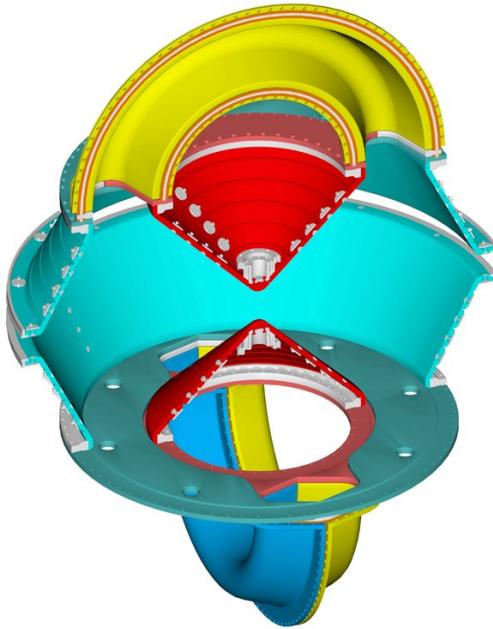


Figure 5: The cut-away view of the HIT-SI experiment. The injectors are attached to the axial ends of the machine. The various colors on the shell indicate components of the shell that are electrostatically insulated from one another. Insulating gaps are needed to allow oscillating fields to enter the equilibrium volume. The cross sectional geometry is a bowtie shape with closed fitting walls. Close fitting walls stabilize the spheromak against a tilt-mode. The shortened geometric axis increases the operational β of the configuration [?].

Figure 6: Total cpu time (in μs) spent in the three core stages of the NIMROD time-advance. For both figures, the vertical and horizontal axes represent the natural logarithm of the total number of cpu time and the natural logarithm of the number of processors ($nprocs$) used in each simulation. The black traces correspond to group 1 and the red traces to group 2. Both metrics indicate `get_rhs` calculations to be most computationally-intensive part of the simulations. Note that `get_rhs` is very sensitive to the load balance per processor. The computation time in all three components linearly scale with $nprocs$ in a log-log plot as one would expect.

0.4 QMCPACK

0.4.1 Introduction

QMCPACK, open-source QMC package, implements state-of-art QMC algorithms. Among them are the improved optimization algorithms based on the energy minimization method of Umrigar et al. (C. J. Umrigar et al. 2007) and the fast multi-determinant method by Clark et al.(Clark et al. 2011). QMCPACK makes extensive use of object-oriented and generic programming and design patterns for reusability and extensibility(J. Kim 2010). High computational efficiency is achieved through inlined specializations of C++ templates and by using explicit single instruction, multiple data (SIMD) intrinsics for core kernels. It utilizes standard file formats for input and output, using XML standard and HDF5, to streamline the QMC workflow and facilitate data exchange with other electronic structure applications.

0.4.2 Background and Motivation

Many methods fall under the broad heading of quantum Monte Carlo. We consider the two continuum methods perhaps most widely used: variational Monte Carlo (VMC) and diffusion Monte Carlo (DMC). These methods are stochastic means to solve the Schrödinger equation, an eigenvalue equation of the form,

$$(37) \quad \hat{\mathcal{H}}\Psi = E\Psi,$$

where $\hat{\mathcal{H}}$ is the *Hamiltonian*, or total energy operator. The Hamiltonian depends on the type of system studied and describes the interactions between particles. In this work, we consider a system of electrons and ionized atomic cores, so that

$$(38) \quad \hat{\mathcal{H}} = -\frac{\hbar^2}{2m}\nabla^2 + \sum_{i < j \in \text{elecs}} \frac{e^2}{r_{ij}} + \sum_{I \in \text{ions}} \hat{V}_I^{\text{SL}},$$

where \hat{V}_I^{SL} is a *pseudopotential* operator. For a system containing N electrons, we use the notation that \mathbf{R} is a $3N$ -dimensional vector representing the positions of the electrons.

The most commonly used trial wave function for solid state systems takes the Slater-Jastrow type,

$$(39) \quad \Psi_T = \det \left[\phi_n^\uparrow(\mathbf{r}_j^\uparrow) \right] \det \left[\phi_n^\downarrow(\mathbf{r}_j^\downarrow) \right] e^{J_1(\mathbf{R}; \mathbf{I})} e^{J_2(\mathbf{R})},$$

where ϕ_n are single-particle orbitals, and J_1 and J_2 are one-body and two-body Jastrow correlation factors. Here, $\mathbf{I} = \{\mathbf{i}_1 \dots \mathbf{i}_M\}$ is the $3M$ -dimensional vector giving the coordinates of the nuclei or ions. In the absence of magnetic fields, electrons are assigned a definite spin and the configuration is given as

$$(40) \quad \mathbf{R} = \{\mathbf{r}_1^\uparrow \dots \mathbf{r}_{N^\uparrow}^\uparrow, \mathbf{r}_1^\downarrow \dots \mathbf{r}_{N^\downarrow}^\downarrow\}.$$

Throughout this paper, we will use both the \mathbf{r} and the $\mathbf{r}^\uparrow/\mathbf{r}^\downarrow$ notations, where appropriate.

0.4.3 Science Driver for Metric Problem

Throughout the year, we will perform QMC total energy calculations of bulk systems of 100-500 electrons. The target problems represent the typical workloads of QMCPACK on petascale computers, especially those with GPUs. While each problem differs in details, e.g., pseudopotentials, grid spacing etc, the QMC workflow is identical: i) execute density functional theory calculations with a large number of k-points; ii) optimize the trial wavefunction; and iii) perform diffusion Monte Carlo calculations to reach < 1 mHa error in the total energy. The primary metric is the efficiency of step iii) which consumes more than 90% of the computing resources.

The first benchmark problem is a graphite system consisting of 64 carbon atoms (256 electrons), namely $\text{gr}(4 \times 4 \times 1)$. The trial wavefunction is already optimized and only step iii) will be executed for benchmarks. Note that the performance limiting factors of ii) are the same as those of iii) and any improvement will be directly translated into the efficiency gains of ii) step. The input single-particle orbitals allow us to study several problem sizes as denoted by $\text{gr}(N_{xy} \times N_{xy} \times N_z)$ with $N_{xy} = 3, 4$, and 6 and $N_z = 1$ and 2 and they are used to obtain an extrapolated QMC energy at the infinite limit.

The efficiency of a QMC calculation is measured by the generation rate of Monte Carlo (MC) samples. This excludes the initialization and I/O costs. The total computational efficiency will be measured by the total run time and reducing the total run time, i.e., the time-to-solution of QMC step iii), is the goal of our Joule project.

0.4.4 Q2 Baseline Problem Results

During Q2, we have ported QMCPACK on Titandev (Cray XK6) at OLCF. The build environments and processes are given in Appendix. The baseline performance of $\text{gr}(4 \times 4 \times 1)$ using CPU and

GPU is established with the code from the developers' repository as March/28 2012. Table 13 lists the efficiency for weak scaling runs measured on Titan/TitanDev at OLCF and Monte Rosa at CSCS.

nodes	Titandev					Titan	Monte Rosa
	16	32	64	128	256	8	4
16	763.14	1261.50	1798.13	2094.11	2337.74	604.27	1236.78
32	1532.64	2516.12	3584.13	4258.07	4588.59	1201.28	2473.21
64	2974.91	4987.67	7197.23	8577.73	9453.09	2393.02	4957.18
128	5992.89	9934.58	14247.44	16698.53	18733.51	772.06	
256	11922.92	19800.34	28341.28	33528.28	36165.23	417.66	
512	24045.11	39476.13	56465.09	64951.47	73229.37	8891.37	

Table 13: Efficiency, number MC samples generated per sec, on Titandev (XK6-GPU), Titan (XK6-CPU) and Monte Rosa (XK6). The target number of walkers are varied on Titandev, while they are fixed at 8 (4) on Titan (Monte Rosa). The CPU runs used OMP_NUM_THREADS=8 and 2 (4) MPI tasks per node on Titan (Monte Rosa).

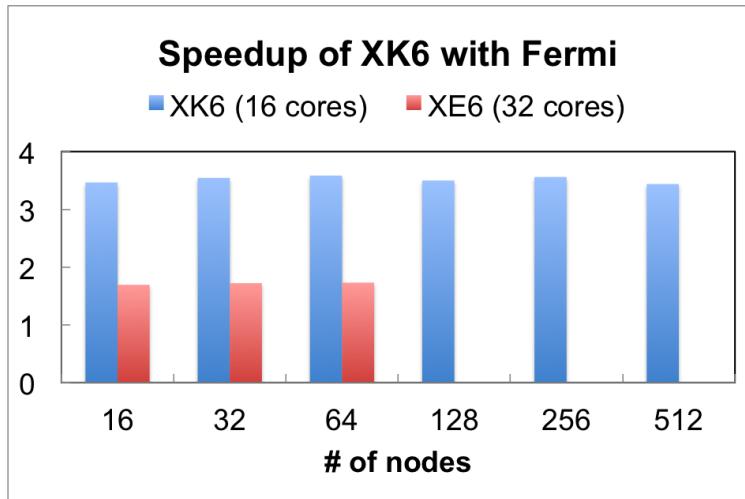


Figure 7: Speedup of Titandev (XK6 with GPU) over Titan (XK6-CPU) and Monte Rosa (XE6). The target number of walkers per node is fixed and 128 for all the runs.

Q3 Goals

They are

- Detailed profiling of gr($4 \times 4 \times 1$) runs and identify performance bottlenecks for both hybrid software and hardware modes

- Tune QMCPACK on AMD interlagos
- Optimize I/O and remove blocking operations on GPUs
- Design hybrid codes to utilize both CPUs and GPUs

Appendices

0.5 A. Software Libraries

0.5.1 Modules Available on the Target Architectures

Q2 Modules: *jaguarpf.ccs.ornl.gov*

It is noted that the module list also includes the software for the Titan development system, essentially a 96 cabinet deployment of the future system and described at the beginning of this document.

```
----- /opt/cray/gem/modulefiles -----
dmapp/3.2.1-1.0400.3965.10.12.gem(default) pmi/3.0.0-1.0000.8661.28.2807.gem(default)
dvs/0.9.0-1.0400.1215.7.13.gem(default) rca/1.0.0-2.0400.30002.5.19.gem(default)
gni-headers/2.1-1.0400.3906.5.1.gem(default) udreg/2.3.1-1.0400.3911.5.6.gem(default)
krca/1.0.0-2.0400.29855.4.28.gem(default) ugni/2.3-1.0400.3912.4.29.gem(default)
pmi/2.1.4-1.0000.8596.8.9.gem xpmem/0.1-2.0400.29883.4.6.gem(default)

----- /opt/cray/xt-asyncpe/default/modulefiles -----
craype-accel-nvidia20 craype-hugepages64M xtpe-istanbul xtpe-shanghai
craype-hugepages128K craype-hugepages8M xtpe-mc12 xtpe-target-native
craype-hugepages16M xtpe-accel-nvidia20 xtpe-mc8
craype-hugepages2M xtpe-barcelona xtpe-network-gemini
craype-hugepages512K xtpe-interlagos xtpe-network-seastar

----- /opt/cray/modulefiles -----
atp/1.3.0 perfetto/5.3.0(default)
atp/1.4.0 perfetto/5.3.1
atp/1.4.1(default) petsc/3.1.09(default)
atp/1.4.2 petsc/3.2.00
atp/1.4.3 petsc-complex/3.1.09(default)
fftw/2.1.5.2 petsc-complex/3.2.00
fftw/2.1.5.3 stat/1.1.3
fftw/3.3.0.0(default) stat/1.2.0.1
ga/4.3.5 stat/1.2.1.1(default)
ga/5.0.1 tpsl/1.0.01
ga/5.0.2(default) tpsl/1.1.01(default)
gdb/7.2(default) tpsl/1.2.00
hdf5/1.8.6 trilinos/10.6.4.0(default)
hdf5/1.8.7(default) trilinos/10.8.3.0
hdf5-parallel/1.8.6 xt-l gdb/1.4
hdf5-parallel/1.8.7(default) xt-libsci/11.0.02
iobuf/2.0.2(default) xt-libsci/11.0.03
libfast/1.0.9 xt-libsci/11.0.04
libsci_acc/1.0.01 xt-libsci/11.0.04.4(default)
libsci_acc/1.0.02(default) xt-libsci/11.0.05
netcdf/4.1.2 xt-libsci/11.0.06
netcdf/4.1.3(default) xt-mpich2/5.3.3
netcdf-hdf5parallel/4.1.2 xt-mpich2/5.3.4
netcdf-hdf5parallel/4.1.3(default) xt-mpich2/5.4.0
ntk/1.3.0 xt-mpich2/5.4.1(default)
ntk/1.4.0 xt-mpich2/5.4.2
ntk/1.5.0(default) xt-mpich2/5.4.3
onesided/1.3.0 xt-mpich2/5.4.4
onesided/1.4.0 xt-shmem/5.3.3
onesided/1.5.0(default) xt-shmem/5.3.4
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
papi/4.1          xt-shmem/5.4.0
papi/4.1.4        xt-shmem/5.4.1(default)
papi/4.2.0(default) xt-shmem/5.4.2
papi/4.3.0        xt-shmem/5.4.3
parallel-netcdf/1.2.0(default) xt-shmem/5.4.4
perf-tools/5.2.3

----- /opt/modules/3.2.6.6/modulefiles -----
dot      module-info null
module-cvs modules-3.2.6.6 use.own

----- /opt/modulefiles -----
PrgEnv-cray/4.0.30(default)      gcc/4.6.2(default)
PrgEnv-gnu/4.0.30(default)       intel/11.1.073
PrgEnv-intel/4.0.30(default)    intel/12.0.5.220
PrgEnv-pgi/4.0.30(default)      intel/12.1.1.256(default)
acml/4.4.0(default)             java/jdk1.6.0_24(default)
acml/5.0.0                  java/jdk1.7.0_02
acml/5.1.0                  java/jdk1.7.0_03
cce/7.4.3                   modules/3.2.6.6
cce/8.0.0                   mrnet/3.0.0(default)
cce/8.0.0.119                netcdf/3.6.2
cce/8.0.0.129                petsc/3.1.09(default)
cce/8.0.1(default)            petsc/3.2.00
cce/8.0.2                   petsc-complex/3.1.09(default)
cce/8.0.2.101                petsc-complex/3.2.00
cce/8.0.3                   pgi/11.10.0
chapel/1.3.0                 pgi/11.10.0.patch
chapel/1.4.0(default)          pgi/11.8.0
cuda/4.0.17                  pgi/11.9.0
cuda/4.0.17a(default)          pgi/12.1.0(default)
cudasdk/4.0.17.0(default)     pgi/12.2.0
cudatools/4.0.17              xe-sysroot/4.0.30
cudatools/4.0.17a(default)    xe-sysroot/4.0.30.securitypatch.20110928
eswrap/1.0.9(default)          xt-asyncpe/5.02
fftw/2.1.5.2                 xt-asyncpe/5.03
fftw/2.1.5.3                 xt-asyncpe/5.04
fftw/3.3.0.0(default)          xt-asyncpe/5.05(default)
gcc/4.4.4                     xt-asyncpe/5.06
gcc/4.5.3                     xt-asyncpe/5.07
gcc/4.6.1                     xt-asyncpe/5.08

----- /sw/xk6/modulefiles -----
DefApps           ncl/6.0.0(default)
adios/1.3.1(default) ncview/1.93g(default)
altd/1.0          netcdf/3.6.2
atlas/3.8.4       netcdf/4.1.3(default)
boost/1.44.0      nfft/3.1.3
cdo/1.5.3         nwchem/6.0
cmake/2.8.6(default) nwchem/6.1
cula-dense/R13a   p-netcdf/1.1.1
ddt/3.0-17491    p-netcdf/1.2.0
ddt/3.1-20638(default) parmetis/4.0.2
dyninst/7.0.1(default) perfexpert/2.0
esmf/5.2.0-p1(default) pspline/1.0
esmf/5.2.0r_with-lapack+netcdf_0 python/2.7.2
esmf/5.2.0rp1     r/2.14.0
git/1.7.8         ruby/1.9.3
globus/5.0.4(default) silo/4.8
gromacs/4.5.5    sprng/2.0b
gsl/1.15          starccm/5.06.010
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
hdf5/1.8.7(default)          starccm/6.06.011(default)
hmpc/2.5.2                   subversion/1.6.17
hmpc/2.5.4                   szip/2.1
hmpc/3.0.0                   tau/2.21_openmp(default)
hmpc/3.0.3                   tau/2.21_pthread
hmpc/3.0.5                   tcl_tk/8.5.8
hmpc-wizard-perfanalyzer/2.0 uduunits/2.1.24
hpctoolkit/5.1.0             vampir/7.5.1-120124-chester
lammps/2011-07-01            vampir/7.5.2-120302(default)
lammps/2012-01-25(default)   vampirtrace/5.12ornl(default)
lammps/2012-02-17            vampirtrace/5.12ornl-chester
lustredu/1.0(default)         vampirtrace/5.13ornl
magma/1.1                    vasp/4.6
marmot/2.4.0(default)        vasp5/5.2
matlab/7.13                  vim/7.3
mercurial/1.9.2              vmd/1.8.7
mxml/2.6                     vmd/1.9
namd/2.7b4                   vtk/5.8
namd/2.8                     wgrib/1.8.0.13b
ncl/5.2.1
```

Q2 Modules: *hopper.nersc.gov*

```
----- /opt/cray/gem/modulefiles -----
dmapp/2.2-1.0301.2791.5.1.gem      rca/1.0.0-2.0301.21810.10.5.gem
dmapp/3.0-1.0301.2968.22.24.gem    rca/1.0.0-2.0301.22737.13.3.gem
dmapp/3.2.1-1.0400.3965.10.12.gem  rca/1.0.0-2.0301.23101.14.211.gem
dmapp/3.2.1-1.0400.3965.10.63.gem(default) rca/1.0.0-2.0301.23291.6.22.gem
dvs/0.9.0-1.0301.1088.58.32.gem    rca/1.0.0-2.0301.23291.6.54.gem
dvs/0.9.0-1.0400.1215.7.13.gem    rca/1.0.0-2.0301.24564.5.18.gem
dvs/0.9.0-1.0400.1215.7.71.gem(default) rca/1.0.0-2.0301.24564.5.4.gem
gni-headers/2.1-1.0301.2792.5.1.gem rca/1.0.0-2.0400.30002.5.19.gem
gni-headers/2.1-1.0301.2931.19.1.gem rca/1.0.0-2.0400.30002.5.75.gem(default)
gni-headers/2.1-1.0400.3906.5.1.gem udreg/2.1-1.0301.2797.5.2.gem
gni-headers/2.1-1.0400.4156.6.1.gem(default) udreg/2.2-1.0301.2966.16.2.gem
krca/1.0.0-2.0400.29855.4.28.gem  udreg/2.3.1-1.0400.3911.5.13.gem(default)
krca/1.0.0-2.0400.30688.5.27.gem(default) udreg/2.3.1-1.0400.3911.5.6.gem
pmi/1.0-1.0000.7901.22.1.gem      ugni/2.1-1.0301.2798.5.2.gem
pmi/1.0-1.0000.8160.39.2.gem      ugni/2.1-1.0301.2967.10.23.gem
pmi/1.0-1.0000.8256.50.6.gem      ugni/2.3-1.0400.3912.4.29.gem
pmi/2.1.1-1.0000.8296.10.8.gem    ugni/2.3-1.0400.4127.5.20.gem(default)
pmi/2.1.2-1.0000.8396.13.5.gem    xpmem/0.1-2.0301.24575.5.2.gem
pmi/2.1.3-1.0000.8574.0.0.gem    xpmem/0.1-2.0301.25333.20.2.gem
pmi/2.1.4-1.0000.8596.8.9.gem    xpmem/0.1-2.0400.29883.4.6.gem
pmi/3.0.0-1.0000.8653.27.8.gem  xpmem/0.1-2.0400.30792.5.6.gem(default)
pmi/3.0.0-1.0000.8661.28.2807.gem(default)
```



```
----- /opt/cray/xt-asyncpe/default/modulefiles -----
craype-accel-nvidia20 craype-hugepages64M  xtpe-interlagos-cu      xtpe-network-seastar
craype-hugepages128K  craype-hugepages8M   xtpe-istanbul       xtpe-shanghai
craype-hugepages16M   xtpe-accel-nvidia20  xtpe-mc12          xtpe-target-native
craype-hugepages2M    xtpe-barcelona     xtpe-mc8           xtpe-xeon
craype-hugepages512K  xtpe-interlagos   xtpe-network-gemini
```



```
----- /opt/cray/modulefiles -----
atp/1.0.2               perfetto/5.3.0(default)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
atp/1.0.3          petsc/3.0.0.9
atp/1.1.0          petsc/3.1.04
atp/1.1.1          petsc/3.1.05
atp/1.1.2          petsc/3.1.08
atp/1.3.0          petsc/3.1.09
atp/1.4.0          petsc/3.2.00(default)
atp/1.4.1          petsc-complex/3.0.0.9
atp/1.4.2(default) petsc-complex/3.1.04
fftw/2.1.5.2       petsc-complex/3.1.05
fftw/2.1.5.3(default) petsc-complex/3.1.08
fftw/3.2.2.1       petsc-complex/3.1.09
fftw/3.3.0.0       petsc-complex/3.2.00(default)
ga/4.3.1           stat/1.1.3
ga/4.3.2           stat/1.2.0.1
ga/4.3.3           stat/1.2.1.1(default)
ga/4.3.5           tpsl/1.0.0
ga/5.0.1           tpsl/1.1.00
ga/5.0.2(default)  tpsl/1.1.01
gdb/7.2(default)   tpsl/1.2.00(default)
hdf5/1.8.4.1      trilinos/10.2.0
hdf5/1.8.5.0      trilinos/10.6.0
hdf5/1.8.6        trilinos/10.6.2.0
hdf5/1.8.7(default) trilinos/10.6.4.0
hdf5-parallel/1.8.4.1 trilinos/10.8.3.0(default)
hdf5-parallel/1.8.5.0 xt-lgdb/1.4(default)
hdf5-parallel/1.8.6 xt-libsci/10.4.9
hdf5-parallel/1.8.7(default) xt-libsci/10.5.01
iobuf/2.0.0        xt-libsci/11.0.01
iobuf/2.0.1        xt-libsci/11.0.02
iobuf/2.0.2(default) xt-libsci/11.0.03
libfast/1.0.7      xt-libsci/11.0.04
libfast/1.0.8      xt-libsci/11.0.05(default)
libfast/1.0.9(default) xt-mpich2/5.1.2
netcdf/3.6.2       xt-mpich2/5.3.0
netcdf/4.0.1.3     xt-mpich2/5.3.2
netcdf/4.1.1.0     xt-mpich2/5.3.3
netcdf/4.1.2       xt-mpich2/5.3.4
netcdf/4.1.3(default) xt-mpich2/5.4.0
netcdf-hdf5parallel/4.0.1.3 xt-mpich2/5.4.1
netcdf-hdf5parallel/4.1.1.0 xt-mpich2/5.4.2
netcdf-hdf5parallel/4.1.2 xt-mpich2/5.4.3
netcdf-hdf5parallel/4.1.3(default) xt-mpich2/5.4.4(default)
ntk/1.0.0          xt-mpt/5.1.2
ntk/1.3.0          xt-mpt/5.3.0
ntk/1.4.0          xt-mpt/5.3.2
ntk/1.5.0(default) xt-mpt/5.3.3(default)
onesided/1.0.0     xt-mpt/5.3.4
onesided/1.3.0     xt-shmem/5.1.2
onesided/1.4.0     xt-shmem/5.3.0
onesided/1.5.0(default) xt-shmem/5.3.2
papi/4.1           xt-shmem/5.3.3
papi/4.1.3         xt-shmem/5.3.4
papi/4.1.4         xt-shmem/5.4.0
papi/4.2.0(default) xt-shmem/5.4.1
parallel-netcdf/1.2.0(default) xt-shmem/5.4.2
perf-tools/5.2.2   xt-shmem/5.4.3
perf-tools/5.2.3   xt-shmem/5.4.4(default)

----- /opt/modulefiles -----
PrgEnv-cray/3.1.61  pathscale/4.0.12.1
PrgEnv-cray/4.0.30  pathscale/4.0.9(default)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
PrgEnv-cray/4.0.36(default)          petsc/3.0.0.9
PrgEnv-gnu/3.1.61                   petsc/3.1.04
PrgEnv-gnu/4.0.30                   petsc/3.1.05
PrgEnv-gnu/4.0.36(default)          petsc/3.1.08
PrgEnv-intel/3.1.61                 petsc/3.1.09
PrgEnv-intel/4.0.30                 petsc/3.2.00(default)
PrgEnv-intel/4.0.36(default)         petsc-complex/3.0.0.9
PrgEnv-pathscale/3.1.61             petsc-complex/3.1.04
PrgEnv-pathscale/4.0.30             petsc-complex/3.1.05
PrgEnv-pathscale/4.0.36(default)    petsc-complex/3.1.08
PrgEnv-pgi/3.1.61                  petsc-complex/3.1.09
PrgEnv-pgi/4.0.30                  petsc-complex/3.2.00(default)
PrgEnv-pgi/4.0.36(default)          pgi/11.0.0
acml/4.4.0(default)                 pgi/11.10.0
apprentice2-desktop/5.1.1           pgi/11.7.0
apprentice2-desktop/5.1.2(default)   pgi/11.9.0(default)
cce/7.3.3                          pgi/12.1.0
cce/7.4.0                          torque/2.4.12
cce/7.4.2                          torque/2.4.15
cce/7.4.4                          torque/2.4.16
cce/8.0.0                          torque/2.5.9(default)
cce/8.0.1                          xe-sysroot/3.1.61
cce/8.0.2(default)                  xe-sysroot/3.1.61.securitypatch.20110823
chapel/1.2.0                        xe-sysroot/3.1.61.securitypatch.20111005
chapel/1.2.1                        xe-sysroot/4.0.30
chapel/1.3.0                        xe-sysroot/4.0.30.securitypatch.20111005
chapel/1.4.0(default)                xe-sysroot/4.0.36
eswrap/1.0.10(default)               xe-sysroot/4.0.36.securitypatch.20111130
fftw/2.1.5.2                         xe-sysroot/4.0.36.securitypatch.20111221
fftw/2.1.5.3(default)                xe-sysroot/4.0.36.securitypatch.20120130
fftw/3.2.2.1                         xt-asyncpe/4.9
fftw/3.3.0.0                         xt-asyncpe/5.00
gcc/4.4.4                           xt-asyncpe/5.01
gcc/4.5.3                           xt-asyncpe/5.02
gcc/4.6.1                           xt-asyncpe/5.03
gcc/4.6.2(default)                  xt-asyncpe/5.04
intel/12.0.4.191                     xt-asyncpe/5.05
intel/12.0.5.220                     xt-asyncpe/5.06
intel/12.1.1.256                     xt-asyncpe/5.07(default)
intel/12.1.2.273(default)            xt-libsci/10.4.9
java/jdk1.6.0_20                     xt-libsci/10.5.01
java/jdk1.6.0_24                     xt-libsci/11.0.01
java/jdk1.7.0_02(default)            xt-libsci/11.0.02
moab/6.0.4                           xt-libsci/11.0.03
moab/6.0.4.local                     xt-libsci/11.0.04
moab/6.1.4(default)                  xt-libsci/11.0.05(default)
moab/6.1.4.local                     xt-papi/4.1.2
modules/3.2.6.6(default)              xt-papi/4.1.3(default)
mrnet/2.2.0.1                         xt-papi/4.1.4
mrnet/3.0.0(default)                  xt-papi/4.2.0
pathscale/3.2.99                     

----- /usr/common/usg/Modules/modulefiles -----
R/2.11.0                            lammps/2011.02
R/2.12.1                            lammps/2011.08
abinit/6.10.3                        lammps/2011.11.09.js
abinit/6.12.1                        lammps/2012.02(default)
abinit/6.4.3(default)                 libtool/2.4(default)
abinit/6.8.2                          libxc/1.0(default)
adios/1.2.1(default)                  mercurial/0.9.5
adios/1.3                            mercurial/1.8.1(default)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
adios/1.3.1          mkl/12.0.5.220
adiosplain/1.2.1(default) mkl/12.1.2.273(default)
asciidoc/8.6.4(default) molpro/2010.1.21
atompaw/3.0.1.3(default) mpiP/3.2.1
autoconf/2.10          mysql/5.0.92(default)
berkeleygw/1.0.1(default) namd/2.7
binutils/2.21.1(default) namd/2.8(default)
bison/2.4              namd/2.8_ccm
boost/1.45             namd/2.8b1
boost/1.47.0(default)  namd/cvs
boost/1.47.0_python2.7.1 nano/2.2.6(default)
cdat/5.2(default)       ncar/5.2.1
cdo/1.4.6(default)      ncar/6.0.0(default)
cfitsio/3.26(default)   nco/4.0.5(default)
cmake/2.8.2(default)   nco/4.0.8
cmake/2.8.7            ncview/1.93g(default)
cp2k/2.1.534(default)  netcdf/4.1.3-nersc
cp2k/2011.10.05        nwchem/6.0(default)
ctss/1.0(default)       openmpi/1.4.3(default)
curl/7.22.0(default)   openmpi/1.4.5
db/dbxml-2.4.16(default) openmpi/1.4.5_mom
ddt/2.6                openss/2.0.1
ddt/3.0                paratec/6.0.0(default)
ddt/3.0-upc            parpack/2.1(default)
ddt/3.1                parpack-gnu/2.1(default)
ddt/3.1-20638(default) pnetcdf/1.2.0(default)
ddt/3.1-21354          pspline/nersc1.0
dfftpack/1.0(default)  python/2.7.1(default)
dlcache/1.0(default)   python/2.7.1-test
dlcache/1.1            scalapack_ccm/2.0.0(default)
espresso/4.2.1(default) siesta/3.0-rc2(default)
espresso/4.3.2          siesta/3.1
espresso/4.3.2-2       silo/4.8(default)
espresso/4.3.2_ccm     sprng/2.0
etsf_io/1.0.3(default) subversion/1.6.4(default)
g09/b1                 subversion/1.7.2
g09/c1(default)         szip/2.1(default)
ga/5.0                 tcl/8.3.3
ga/5.0.3.r8i8          tcl/8.5.9(default)
ga/5.0.i8              totalview/8.7.0
gameSS/1OCT2010R1(default) totalview/8.9.1
gameSS/24MAR2007R6     totalview/8.9.1-1
git/1.7.4              totalview/8.9.2(default)
git/1.7.7.1            totalview-support/1.0.6
git/1.7.7.4(default)   totalview-support/1.1.2(default)
gnuplot/4.4.0           training/1.0(default)
gnuplot/4.4.3(default)  uduunits/2.1.19(default)
gpaw/0.8.0.8092(default) usg/1.1(default)
grace/5.1.22(default)   valgrind/3.6.1
gromacs/4.5.3(default)  vas/4.6.35
gromacs/4.5.5          vas/4.6.35.pkent
gromacs/4.5.5-dp       vas/5.2
gsl/1.14(default)       vas/5.2.11(default)
gsl/1.15               vas/5.2.11.tbdyn
gv/3.7.1(default)       vas/5.2.12
h5part/1.6.2(default)   visit/2.3.2
hdf/4.2r4(default)      visit/2.4.0
hdf5/1.8.7(default)     visit/2.4.1
hdfeos/2.17             visit/2.4.2(default)
hdfeos/5.1.12(default)  vmd/1.9.1(default)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
ipm/2.00(default)          wannier90/1.2(default)
ipm-ccm/2.00(default)      wien2k/11.1_ccm(default)
ipm-openmp/cray/2.00       yambo/3.2.4-r.855(default)
ipm-openmp/pgi/2.00        zlib/1.2.3
ipmio/2.00(default)        zlib/1.2.5(default)
jmol/12.2.12(default)

----- /usr/common/acts/Modules/modulefiles -----
hypre/2.7.0b_0             petsc/3.1_0           slepc/3.1-opkgs_0_complex
hypre/2.7.0b_g             petsc/3.1_0_c++      slepc/3.1-opkgs_g
metis/4.0                  petsc/3.1_0_complex   slepc/3.1-opkgs_g_c++
metis/4.0_0                 petsc/3.1_g          slepc/3.1-opkgs_g_complex
metis/4.0_g                 petsc/3.1_g_c++     slepc/3.1_0
mumps/4.10.0                petsc/3.1_g_complex  slepc/3.1_0_c++
mumps/4.10.0_0              petsc/3.2            slepc/3.1_0_complex
mumps/4.10.0_g              petsc/3.2_0          slepc/3.1_g
parmetis/3.1.1              petsc/3.2_0_c++     slepc/3.1_g_c++
parmetis/3.1.1_0             petsc/3.2_0_complex  slepc/3.1_g_complex
parmetis/3.1.1_g             petsc/3.2_g          superlu/4.0
petsc/#3.1_0_complex#       petsc/3.2_g_c++     superlu/4.0_0
petsc/3.1                  petsc/3.2_g_complex  superlu/4.0_g
petsc/3.1-opkgs_0            scalapack/1.8.0      superlu_dist/2.5
petsc/3.1-opkgs_0_c++        scalapack/1.8.0_0    superlu_dist/2.5_0
petsc/3.1-opkgs_0_complex   scalapack/1.8.0_g     superlu_dist/2.5_g
petsc/3.1-opkgs_g            slepc/3.1            tau/2.21.2(default)
petsc/3.1-opkgs_g_c++        slepc/3.1-opkgs_0   petsc/3.1-opkgs_g_complex
petsc/3.1-opkgs_g_complex   slepc/3.1-opkgs_0_c++
```



```
----- /usr/common/nsg/Modules/modulefiles -----
cfengine/2.1.20(default)    nsg/1.0.0           vim/7.1(default)
ddd/3.3.11(default)         nsg/1.1.0(default)
gpfs/1.0.0(default)         rapidsvn/0.9.8(default)
```



```
----- /usr/common/ftg/Modules/modulefiles -----
bupc/2.14.0(default)        bupc/2.14.0-4.0.36-gnu-4.6.1
bupc/2.14.0-3.1.61-cray-7.4.2 bupc/2.14.0-4.0.36-gnu-4.6.2
bupc/2.14.0-3.1.61-gnu-4.6.1 bupc/2.14.0-4.0.36-intel-12.0.5.220
bupc/2.14.0-3.1.61-pathscale-3.2.99 bupc/2.14.0-4.0.36-intel-12.1.2.273
bupc/2.14.0-3.1.61-pgi-11.7.0 bupc/2.14.0-4.0.36-pathscale-3.2.99
bupc/2.14.0-4.0.36-cray-7.4.4 bupc/2.14.0-4.0.36-pgi-11.9.0
bupc/2.14.0-4.0.36-cray-8.0.2 hpctoolkit
```



```
----- /usr/common/graphics/Modules/modulefiles -----
ParaView/3.10.0  ParaView/3.14.0  libtiff/3.9.5    sdl/1.2.14
ParaView/3.12.0  ferret/6.71     povray/3.7.0.RC3
```

0.6 Machine Event Data Collection

The hardware environments and software interactions with these environments contain important information that often assists in the process of enhancing for scaling or efficiency. This year we will execute MPI, MPI + OpenMP, MPI + OpenMP + CUDA programs. To capture the machine perspective of binary execution for the parallelism supported in these models requires that our applications are instrumented to gather the desired information. There are different tools that exist to assist us in this process that range from automatically to user instrumented.

The chipsets in use for all host computations this FY support the 'caliper' approach to profiling. In this approach one reads the event count before and after a performance-critical region of code and the before-count is subtracted from the after-count to yield the number of events that occurred between the reads. So, if a code region has more than a single work phase, then one has to begin and end profiling of the distinct phases at the entry and exit points of these phases respectively if the intent is to separate the costs of the phases. This makes sense as the clocks / counters are continuously running regardless of which phase is in execution and cannot distinguish these phases placing the responsibility on the instrumenter. The processors in use include support for two methods of monitoring processor performance: performance monitor counters and instruction based sampling (IBS). There are two types of performance counters -located in each core of each compute unit and the Northbridge monitor counters. The core of each compute core has six 48-bit performance counters, while a node has four 48-bit performance counters to monitor Northbridge activity. They are not all equally useful to us. The instruction execution performance is profiled by tagging one micro-op associated with an instruction. Instructions that decode to more than one micro-op return different performance data depending upon which micro-op associated with the instruction is tagged. These micro-ops are associated with the RIP of the next instruction to retire. We are interested in the data cache access status associated with the micro-op and are currently tracking the number of L2 data cache misses.

0.6.1 KRP Machine Event Data Collection API

The routines krp_init(1), krp_rpt_init(2), krp_rpt_init_sum(4), krp_init_sum(3), krp_rpt(6) are hacked together to give the application users quick access to useful information; krp_stat(5) does not touch the counters rather performs some statistics over measured observables. The routines can be easily modified to suit the needs of the user and currently depend on PAPI semantics designed to access relevant event counters on the targets. Also, clearly $2^{48} = 281474976710656 \sim O(1E14)$ and as such we depend on this software to mitigate overflow issues in the counters.

1 krp_init()

```
function VOID KRP_INIT(int iam , int * hw_counters , long long int * rcy , long long int * rus ,
long long int * ucy , long long int * uus)
```

int iam process id of calling process

int * hw_counters initialized on return to the number of hardware counters found on the chip

long long int * rcy initialized on return to the current reading of the real cycle count

long long int * rus initialized on return to the current value of the system clock in real microseconds

long long int * ucy initialized on return to the current reading of the virtual cycle count

long long int * uus initialized on return to the current value of the system clock in virtual microseconds

The routine is called one time at the point in the software application by all processes that wish to start monitoring the clocks and the machine event counters. The routine explicitly requests event counts on total retired instructions, total floating point operations, and total number of L2 data cache misses -this can be changed but is what I decided for now. The real time clocks run all the time. The virtual readings account for cycles / time spent only in user code. Once called, the machine event counters for the 3 events are activated. Some information about the node architecture is reported by process with id = 0. Non-blocking call.

end function

0.6.2 KRP Use Examples - Level 3 BLAS

For now, just consider matrix multiplication as a prototype problem. The basic kernel be will stated symbolically, and the time and storage complexity estimated. Instead of time, the operation complexity is estimated. The operation complexity can be turned into a time with knowledge of the specifics of the processor computing the kernels. For instance, if a processor can compute $X[\frac{FP_Ops}{cycle}]$ and has clock frequency of $Y[\frac{cycles}{second}]$, then the estimated time to compute kernel W (ignoring details of memory capacity and structure, time spent fetching and storing operands, etc.) is $time(W)[seconds] := \frac{N[FP_Ops(W)]}{X[\frac{FP_Ops}{cycle}]Y[\frac{cycles}{second}]}.$

Matrix-Matrix Multiplication. The operation is $C \leftarrow \alpha AB + \beta C$ where $A \in \mathbb{C}^{m,l}$, $B \in \mathbb{C}^{l,n}$ $C \in \mathbb{C}^{m,n}$ and α, β are complex scalars. Clearly, the operation includes $ml + ln + mn + 2$ complex numbers to be stored -not including any temporary work buffers in the implementation. In general, we write $[\forall i = 1 \dots m][\forall j = 1, \dots n][C_{i,j} = \alpha(\sum_{k=1}^{k=l} A_{i,k}B_{k,j}) + \beta C_{i,j}]$. The cost of $T_1 \sim \alpha(\sum_{k=1}^{k=l} A_{i,k}B_{k,j}) =$

2 krp_rpt_init()

```
function VOID KRP_RPT_INIT(int iam , MPI_Comm comm , int * hw_counters , long long int * rcy , long long int * rus , long long int * ucy , long long int * uus , char * bf)
```

int iam process id of calling process

MPI_Comm comm MPI communicator of which process iam is a member

int * hw_counters used and initialized on return to the number of hardware counters found on the chip

long long int * rcy used and initialized on return to the current reading of the real cycle count

long long int * rus used and initialized on return to the current value of the system clock in real microseconds

long long int * ucy used and initialized on return to the current reading of the virtual cycle count

long long int * uus used and initialized on return to the current value of the system clock in virtual microseconds

char * bf character buffer that is truncated to 128 CHARs internally -allows the user to label a measurement reporting event

The routine is called at any point in the software application after the init routine to report (rpt) the change in the values of the machine event counters and the real / virtual time / cycle readings off the clock. The routine explicitly requests event counts on total retired instructions, total floating point operations, and total number of L2 data cache misses. High and low values, and the standard deviation in the values of events over processes in the communicator are reported. The counters are stopped to make this analysis. The current clock values are assigned, and the event counters restarted prior to returning control to the calling processes. Blocking call.

end function

3 krp_init_sum()

```
function VOID KRP_INIT_SUM(int iam , int * hw_counters , long long int * rcy , long long int * rus , long long int * ucy , long long int * uus , long int * rt_rus , long int * rt_ins , long int * rt_fp , long int * rt_dcm)
```

int iam process id of calling process

int * hw_counters used and initialized on return to the number of hardware counters found on the chip

long long int * rcy used and initialized on return to the current reading of the real cycle count

long long int * rus used and initialized on return to the current value of the system clock in real microseconds

long long int * ucy used and initialized on return to the current reading of the virtual cycle count

long long int * uus used and initialized on return to the current value of the system clock in virtual microseconds

long int * rt_rus variable used to aggregate the value of the real microseconds for the calling process; the change in rus between subsequent calls is added to this variable on return

long int * rt_ins variable used to aggregate the value of the retired instructions for the calling process; the change in INS between subsequent calls is added to this variable on return

long int * rt_fp variable used to aggregate the value of the floating point operations for the calling process; the change in FP_OP between subsequent calls is added to this variable on return

long int * rt_dcm variable used to aggregate the value of the L2 data cache misses for the calling process; the change in L2DCM between subsequent calls is added to this variable on return

The routine is called at any point in the software application after the init routine to aggregate clock values and machine events for the calling process. It is useful to measure the expense of computing distinct work phases in loop regions of the codes, for instance. The routine explicitly requests aggregation of total retired instructions, total floating point operations, total number of L2 data cache misses, and real microseconds since the previous call. The counters are stopped to update the values. The current clock / cycle values are assigned, and the event counters restarted prior to returning control to the calling processes. Non-blocking call.

end function

4 krp_rpt_init_sum()

function VOID KRP_RPT_INIT_SUM(int iam , MPI_Comm comm , int * hw_counters , long long int * rcy , long long int * rus , long long int * ucy , long long int * uus , long int * rt_rus , long int * rt_ins , long int * rt_fp , long int * rt_dcm , char * bf)

int iam process id of calling process

MPI_Comm comm MPI communicator of which process iam is a member

int * hw_counters used and initialized on return to the number of hardware counters found on the chip

long long int * rcy used and initialized on return to the current reading of the real cycle count

long long int * rus used and initialized on return to the current value of the system clock in real microseconds

long long int * ucy used and initialized on return to the current reading of the virtual cycle count

long long int * uus used and initialized on return to the current value of the system clock in virtual microseconds

long int * rt_rus variable used to aggregate the value of the real microseconds for the calling process; the change in rus between subsequent calls is added to this variable on return

long int * rt_ins variable used to aggregate the value of the retired instructions for the calling process; the change in INS between subsequent calls is added to this variable on return

long int * rt_fp variable used to aggregate the value of the floating point operations for the calling process; the change in FP_OP between subsequent calls is added to this variable on return

long int * rt_dcm variable used to aggregate the value of the L2 data cache misses for the calling process; the change in L2DCM between subsequent calls is added to this variable on return

char * bf character buffer that is truncated to 128 CHARs internally -allows the user to label a measurement reporting event

The routine is called at any point in the software application after the init routine to report (rpt) the change in the values of the machine event counters and the real / virtual time / cycle readings off the clock, and to aggregate clock values and machine events for the calling process. It is useful to report the expense of computing distinct work phases in loop regions of the codes, for instance. The routine explicitly requests aggregation of total retired instructions, total floating point operations, total number of L2 data cache misses, and real microseconds since the previous call. High and low values, and the standard deviation in the values of events over processes in the communicator are reported. The counters are stopped to update the values and conduct this analysis. The current clock / cycle values are assigned, and the event counters restarted prior to returning control to thegalling processes. Basically the same as the init_sum() routine, but in verbose mode and statistics done for the calling processes. Blocking call.

end function

5 krp_stat()

```
function VOID KRP_STAT(int iam , MPI_Comm comm , long int rt_rus , long int rt_ins , long int rt_fp , long int rt_dcm)
```

int iam process id of calling process

MPI_Comm comm MPI communicator of which process iam is a member

long int rt_rus variable used to aggregate the value of the real microseconds for the calling process

long int rt_ins variable used to aggregate the value of the retired instructions for the calling process

long int rt_fp variable used to aggregate the value of the floating point operations for the calling process

long int rt_dcm variable used to aggregate the value of the L2 data cache misses for the calling process

The routine is called at any point in the software application after an init_sum call to aggregate clock values and machine events for the calling process. It is useful to report the expense of computing distinct work phases in codes that couple MPI processes with lightweight OpenMP or POSIX threads and does not touch the event counters (as such changes their values). The routine aggregates of total retired instructions, total floating point operations, total number of L2 data cache misses, and real microseconds. High and low values, and the standard deviation in the values of events over processes in the communicator are reported. Blocking call.

end function

6 krp_rpt()

```
function VOID KRP_RPT(int iam , MPI_Comm comm , int * hw_counters , long long int * rcy ,  
long long int * rus , long long int * ucy , long long int * uus , char * bf)
```

int iam process id of calling process

MPI_Comm comm MPI communicator of which process iam is a member

int * hw_counters used internally in loop over events

long long int * rcy used internally in forming differences

long long int * rus used internally in forming differences

long long int * ucy used internally in forming differences

long long int * uus used internally in forming differences

char * bf character buffer that is truncated to 128 CHARs internally -allows the user to label a measurement reporting event

The routine is called at any point in the software application after the init routine to report (rpt) the change in the values of the machine event counters and the real / virtual time / cycle readings off the clock. The routine explicitly reports event counts on total retired instructions, total floating point operations, and total number of L2 data cache misses. High and low values, and the standard deviation in the values of events over processes in the communicator are reported. The counters are stopped to make this analysis. Neither clock / cycle values are updated, nor are the events counters restarted prior to return to calling process. Intended to be called at the end of execution. Is a blocking call.

end function

$\alpha(A_{i,1}B_{1,j} + A_{i,2}B_{2,j} + \cdots + A_{i,l}B_{l,j})$ is $l+1$ complex multiplies and $l-1$ complex adds. The cost of $T_2 \sim \beta C_{i,j}$ is 1 complex multiply. The cost of $T_1 + T_2$ is 1 complex add. The total cost of each $C_{i,j}$ element is $l+1+1 = l+2$ complex multiplies and $l-1+1 = l$ complex adds. This cost is incurred mn times to account for each $C_{i,j}$. Thus, if the arithmetic was performed on real numbers, the total complexity is $mn(l+2) = mnl + 2mn$ multiplies and $mn(l)$ adds, or $2mnl + 2mn$ floating point operations.

However, on paper, the cost of a single *complex* multiplication is 5 multiplies + 2 adds, and the cost of a single *complex* add is 2 adds. We have assumed a single multiply for i^2 to be included in the product of two numbers. That is, $(\alpha, \beta)(a, b) = (\alpha + i\beta)(a + ib) = (\alpha a + (-)\beta b) + i(\alpha b + \beta a) = (\alpha a - \beta b, \alpha b + \beta a)$, and $(\alpha, \beta) + (a, b) = (\alpha + i\beta) + (a + ib) = (\alpha + a) + i(\beta + b) = (\alpha + a, \beta + b)$. Circuit based implementations tend to employ 4 multiplies and 2 adds with the knowledge that the terms affiliated with the i^2 will be minus one. (Golub's method is again different w/ 3 multiplies and 3 additions.) Thus, each $C_{i,j}$ executes $(l+2)$ *complex* multiplies \times (4 multiplies + 2 adds) per *complex* multiply or $4(l+2) = 4l+8$ multiplies and $2(l+2) = 2l+4$ adds, and l *complex* adds \times (2 adds) per *complex* add or $2l$ adds. The total complexity is $4l+8$ multiplies and $2l+4+2l = 4l+4$ adds, or $8l+12$ floating point operations.

In summary, including the fact that there are mn total elements, our theoretical estimates of floating point costs for `_gemm()` are in table 14.

	Multiples	Adds	Total
real	$mnl + 2mn$	mnl	$2mnl + 2mn$
complex	$4mnl + 8mn$	$4mnl + 4mn$	$8mnl + 12mn$

Table 14: Theoretical complexity of $C(m, n) \leftarrow \alpha A(m, l)B(l, n) + \beta C(m, n)$.

Profiled User Scenarios The implementation of the same kernel (with known theoretical complexity) in a variety of contrived use scenarios both assists the users in understanding how to instrument their application, and also assists tool development and testing, and code optimization. We present the output of an interactive session on the target for clarity. In each case, the theoretical measure of work is the number of floating point computations required to complete the instance of complex matrix multiplication described. The user scenarios are described in pseudocodes (7,8,9). There are two basic modes of profiling the user scenarios we have executed (PGI,GNU): detailed profiling and simple profiling where internal work phases are either separated or ignored. In both cases, a total picture is measured and returned. For each of the three user scenarios, raw output will be displayed as an example of detailed and simple cases.

7 Two sequential work phases in Loop: calculate $C \leftarrow \alpha AB + \beta C$

Input: $np, m, l, n, m', l', n', nits$

```
get MPI process ID  $ip \in [0, np - 1]$  ;  
initialize  $x, y, z, a, b, c, \alpha, \beta$  ;  
for  $i = 0 \rightarrow nits$  do  
     $zgemm(m, l, n, a, b, c, \alpha, \beta)$  ;  
     $zgemm(m', l', n', x, y, z, \alpha, \beta)$  ;  
end for
```

Modules

```
jaguarpf> module list  
Currently Loaded Modulefiles:  
1) modules/3.2.6.6  
2) xtpe-network-gemini  
3) pgi/12.1.0  
4) xt-libsci/11.0.04.4  
5) udreg/2.3.1-1.0400.3911.5.6.gem  
6) ugni/2.3-1.0400.3912.4.29.gem  
7) pmi/3.0.0-1.0000.8661.28.2807.gem  
8) dmapp/3.2.1-1.0400.3965.10.12.gem  
9) gni-headers/2.1-1.0400.3906.5.1.gem  
10) xpmem/0.1-2.0400.29883.4.6.gem  
11) xt-asyncpe/5.05  
12) atp/1.4.1  
13) PrgEnv-pgi/4.0.30  
15) xtpe-interlagos  
16) lustredu/1.0  
17) DefApps  
18) altd/1.0  
19) papi/4.2.0  
20) torque/2.4.1b1-snap.200905191614  
21) moab/6.1.3  
22) nodestat/2.2-1.0400.29866.4.3.gem  
23) sdb/1.0-1.0400.30000.6.18.gem  
24) MySQL/5.0.64-1.0000.4667.20.1  
25) lustre-crash_gem_s/1.8.4_2.6.32.45_0.3.2_1.0400.6221.1.1-1.0400.30303.0.0novmap1  
26) hss-llm/6.0.0  
27) Base-opts/1.0.2-1.0400.29823.8.1.gem
```

Environment

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
jaguarpf> env
LESSKEY=/etc/lesskey.bin
MODULE_VERSION_STACK=3.2.6.6
INFODIR=/usr/local/info:/usr/share/info:/usr/info
NNTPSERVER=news
PAPI_POST_LINK_OPTS= -L/opt/cray/papi/4.2.0/perf_events/no-cuda/lib -lpapi
HOSTNAME=jaguarpf-ext1
CRAY_UDREG_INCLUDE_OPTS=-I/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/include
RCLOCAL_BASEOPTS=true
PBS_VERSION=TORQUE-2.5.9-snap.201111021154
XKEYSYMDB=/usr/share/X11/XKeysymDB
CRAY_SITE_LIST_DIR=/etc/opt/cray/modules
LIBRARYMODULES=/opt/modules/3.2.6.6/init/.librarymodules:acml:alps:apprentice2:
blcr:dmapp:fftw:ga:gni-headers:hdf5:libfast:mpich1:mrnet:netcdf:ntk:onesided:
petsc:petsc-complex:pmi:portals:rca:tpsl:trilinos:udreg:ugni:xpmem:xt-atp:
xt-lgdb:xt-libsci:xt-mpt:xt-papi:/etc/opt/cray/modules/site_librarymodules
PE_ENV=PGI
PAPI_VERSION=4.2.0
SHELL=/bin/bash
TERM=xterm-color
HOST=jaguarpf-ext1
HISTSIZE=1000
PROFILEREAD=true
XTOS_VERSION=4.0.30
BATCH_ALLOC_COOKIE=0
PBS_JOBNAME=STDIN
SSH_CLIENT=166.250.9.42 4993 22
CRAY_UGNI_POST_LINK_OPTS=-L/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/lib64
CRAY_XPMEM_POST_LINK_OPTS=-L/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/lib64
CRAY_MPICH2_DIR=/opt/cray/mpt/5.4.1/xt/gemini/mpich2-pgi/109
ALTD_SELECT_OFF_USERS=
CRAY_PRGENVPGI=loaded
ALTD_SELECT_ON=0
PBS_ENVIRONMENT=PBS_INTERACTIVE
MORE=-s1
BATCH_JOBID=1060095
ALTD_VERBOSE=0
SSH_TTY=/dev/pts/25
PBS_O_WORKDIR=/tmp/work/roche/kr-tsts/mm-study/f-ex
PBS_TASKNUM=1
USER=roche
ASYNCPE_DIR=/opt/cray/xt-asyncpe/5.05
BUILD_OPTS=/opt/cray/xt-asyncpe/5.05/bin/build-opts
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
INTEL_PRE_COMPILE_OPTS= -msse3
LS_COLORS=
LD_LIBRARY_PATH=/opt/cray/papi/4.2.0/perf_events/no-cuda/lib:
/opt/pgi/12.1.0/linux86-64/12.1/libso:/opt/pgi/12.1.0/linux86-64/12.1/lib:
/opt/cray/atp/1.4.1/lib
PBS_O_HOME=/ccs/home/roche
XNLSPATH=/usr/share/X11/nls
WORKDIR=/tmp/work/roche
PBS_NNODES=64
ENV=/etc/bash.bashrc
MPICH_ABORT_ON_ERROR=1
MPICH_DIR=/opt/cray/mpt/5.4.1/xt/gemini/mpich2-pgi/109
PGI_VERS_STR=12.1.0
ALTD_ON=1
HOSTTYPE=x86_64
ATP_POST_LINK_OPTS=
-Wl,-L/opt/cray/atp/1.4.1/lib/
-Wl,-lAtpSigHandler
-Wl,--undefined=__atpHandlerInstall
RCLOCAL_PRGENV=true
PBS_GPUFILE=/var/spool/torque/aux//1060095.jaguarpf-sys0.ccs.ornl.govgpu
PBS_MOMPORT=15003
FROM_HEADER=
CRAY_MPICH2_VERSION=5.4.1
PE_PRODUCT_LIST=CRAY_LLM:XTPE_INTERLAGOS:CRAY_MPICH2:ATP:ASYNCPE:CRAY_XPMEM:
CRAY_DMAPP:CRAY_PMI:CRAY_UGNI:CRAY_UDREG:LIBSCI:PGI:XT_SYSROOT:PAPI
PAPI_INTERFACE=perf_events
PAGER=less
FFTW_SYSTEM_WISDOM_DIR=/opt/xt-libsci/11.0.04.4
PGI_VERSION=12.1
CSHEDIT=emacs
PAPI_INCLUDE_OPTS= -I/opt/cray/papi/4.2.0/perf_events/no-cuda/include
XDG_CONFIG_DIRS=/etc/xdg
PBS_O_QUEUE=debug
MINICOM=-c on
USERMODULES=/opt/modules/3.2.6.6/init/.usermodules:acml:alps:apprentice2:
atp:blcr:cce:craypat:dmapp:fftw:ga:gcc:gni-headers:hdf5:intel:libfast:
mpich1:mrnet:netcdf:ntk:onesided:pathscale:petsc:petsc-complex:pgi:pmi:
portals:PrgEnv-cray:PrgEnv-gnu:PrgEnv-intel:PrgEnv-pathscale:PrgEnv-pgi:
rca:stat:tpsl:trilinos:udreg:ugni:xpmem:xt-asyncpe:xt-crayspat:xt-lgdb:
xt-libsci:xt-mpt:xt-papi:xt-totalview:/etc/opt/cray/modules/site_usermodules
CRAY_DMAPP_INCLUDE_OPTS=
-I/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/include
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
-I/opt/cray/gni-headers/2.1-1.0400.3906.5.1.gem/include
PGI=/opt/pgi/12.1.0
APRUN_XFER_LIMITS=0
PATH=/opt/cray/l1m/default/bin:/opt/cray/l1m/default/etc:
/opt/cray/lustre-crash_gem_s/1.8.4_2.6.32.45_0.3.2_1.0400.6221.1.1-1.0400.30303.0.0novmap1/sbin:
/opt/cray/lustre-crash_gem_s/1.8.4_2.6.32.45_0.3.2_1.0400.6221.1.1-1.0400.30303.0.0novmap1/bin:
/opt/cray/MySQL/5.0.64-1.0000.4667.20.1/sbin:/opt/cray/MySQL/5.0.64-1.0000.4667.20.1/bin:
/opt/cray/sdb/1.0-1.0400.30000.6.18.gem/bin:/opt/cray/nodestat/2.2-1.0400.29866.4.3.gem/bin:
/opt/moab/default/bin:/opt/torque/default/bin:/opt/cray/papi/4.2.0/perf_events/no-cuda/bin:
/sw/xk6/altd/bin:/sw/xk6/bin:/sw/xk6/lustredu/1.0/sles11.1_gnu4.3.4/lustredu/bin:
/opt/cray/eslogin/eswrap/1.0.9/bin:/opt/cray/atp/1.4.1/bin:/opt/cray/xt-asyncpe/5.05/bin:
/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/bin:/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/bin:
/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/bin:/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/bin:
/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/bin:/opt/pgi/12.1.0/linux86-64/12.1/bin:
/opt/modules/3.2.6.6/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/X11R6/bin:/usr/games:
/opt/bin:/usr/lib/mit/bin:/usr/lib/mit/sbin:/opt/dell/srvadmin/bin:/opt/bin:/opt/public/bin:
/opt/bin:/opt/public/bin
MAIL=/var/mail/roche
MODULE_VERSION=3.2.6.6
SYSROOT_GCC_VER=4.3.2
PAPI_INTERFACE_VERSION=no-cuda
PBS_O_LOGNAME=roche
CPU=x86_64
XTPE_NETWORK_TARGET=gemini
ESWRAP_LOGIN=jaguarpf-login5
PBS_O_LANG=en_US.UTF-8
PBS_JOBCOOKIE=36AED45CD1CC16AA653C005E8CF9FA5D
ASYNCPE_VERSION=5.05
CRAY_UDREG_POST_LINK_OPTS=-L/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/lib64
PWD=/tmp/work/roche/kr-tsts/mm-study/f-ex
INPUTRC=/etc/inputrc
TARGETMODULES=/opt/modules/3.2.6.6/init/.targetmodules:craype-cpu-sandybridge:
craype-network-aries:xtpe-barcelona:xtpe-interlagos:xtpe-istanbul:xtpe-mc12:
xtpe-mc8:xtpe-netork-aries:xtpe-network-gemini:xtpe-network-seastar:xtpe-shanghai:
xtpe-target-cnl:xtpe-target-native:/etc/opt/cray/modules/site_targetmodules
_LMFILES_= /opt/modulefiles/modules/3.2.6.6:
/opt/modulefiles/xe-sysroot/4.0.30.securitypatch.20110928:
/opt/cray/xt-asyncpe/default/modulefiles/xtpe-network-gemini:
/opt/modulefiles/pgi/12.1.0:/opt/cray/modulefiles/xt-libsci/11.0.04.4:
/opt/cray/gem/modulefiles/udreg/2.3.1-1.0400.3911.5.6.gem:
/opt/cray/gem/modulefiles/ugni/2.3-1.0400.3912.4.29.gem:
/opt/cray/gem/modulefiles/pmi/3.0.0-1.0000.8661.28.2807.gem:
/opt/cray/gem/modulefiles/dmapp/3.2.1-1.0400.3965.10.12.gem:
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
/opt/cray/gem/modulefiles/gni-headers/2.1-1.0400.3906.5.1.gem:  
/opt/cray/gem/modulefiles/xpmem/0.1-2.0400.29883.4.6.gem:  
/opt/modulefiles/xe-sysroot/4.0.30:/opt/modulefiles/xt-asyncpe/5.05:  
/opt/cray/modulefiles/atp/1.4.1:/opt/modulefiles/PrgEnv-pgi/4.0.30:  
/opt/cray/modulefiles/xt-mpich2/5.4.1:  
/opt/cray/xt-asyncpe/default/modulefiles/xtpe-interlagos:  
/opt/modulefiles/eswrap/1.0.9:/sw/xk6/modulefiles/lustredu/1.0:  
/sw/xk6/modulefiles/DefApps:/sw/xk6/modulefiles/altd/1.0:  
/opt/cray/modulefiles/papi/4.2.0:  
/opt/modulefiles/torque/2.4.1b1-snap.200905191614:  
/opt/modulefiles/moab/6.1.3:  
/opt/cray/gem/modulefiles/nodestat/2.2-1.0400.29866.4.3.gem:  
/opt/cray/gem/modulefiles/sdb/1.0-1.0400.30000.6.18.gem:  
/opt/cray/modulefiles/MySQL/5.0.64-1.0000.4667.20.1:  
/opt/cray/modulefiles/lustre-crash_gem_s/-  
1.8.4_2.6.32.45_0.3.2_1.0400.6221.1.1-1.0400.30303.0.0novmap1:  
/opt/modulefiles/hss-llm/6.0.0:  
/opt/modulefiles/Base-opts/1.0.2-1.0400.29823.8.1.gem  
PBS_NODENUM=0  
LANG=en_US.UTF-8  
PGROUPD_LICENSE_FILE=/opt/pgi/license.dat  
SYSTEM_USERDIR=/tmp/work/roche  
PYTHONSTARTUP=/etc/pythonstart  
MODULEPATH=/opt/cray/gem/modulefiles:/opt/cray/xt-asyncpe/default/modulefiles:  
/opt/cray/modulefiles:/opt/modules/3.2.6.6/modulefiles:/opt/modulefiles:  
/sw/xk6/modulefiles  
PBS_NUM_NODES=1  
LOADEDMODULES=modules/3.2.6.6:xe-sysroot/4.0.30.securitypatch.20110928:  
xtpe-network-gemini:pgi/12.1.0:xt-libsci/11.0.04.4:udreg/2.3.1-1.0400.3911.5.6.gem:  
ugni/2.3-1.0400.3912.4.29.gem:pmi/3.0.0-1.0000.8661.28.2807.gem:  
dmapp/3.2.1-1.0400.3965.10.12.gem:gni-headers/2.1-1.0400.3906.5.1.gem:  
xpmem/0.1-2.0400.29883.4.6.gem:xe-sysroot/4.0.30:xt-asyncpe/5.05:atp/1.4.1:  
PrgEnv-pgi/4.0.30:xt-mpich2/5.4.1:xtpe-interlagos:eswrap/1.0.9:lustredu/1.0:  
DefApps:altd/1.0:papi/4.2.0:torque/2.4.1b1-snap.200905191614:moab/6.1.3:  
nodestat/2.2-1.0400.29866.4.3.gem:sdb/1.0-1.0400.30000.6.18.gem:  
MySQL/5.0.64-1.0000.4667.20.1:  
lustre-crash_gem_s/1.8.4_2.6.32.45_0.3.2_1.0400.6221.1.1-1.0400.30303.0.0novmap1:  
hss-llm/6.0.0:Base-opts/1.0.2-1.0400.29823.8.1.gem  
SHMEM_ABORT_ON_ERROR=1  
PATHSCALE_PRE_COMPILE_OPTS= -march=barcelona  
CRAY_DMAPP_POST_LINK_OPTS=-L/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/lib64  
PBS_O_SHELL=/bin/bash  
PBS_SERVER=jaguarpf-sys0.ccs.ornl.gov
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
PBS_JOBID=1060095.jaguarpf-sys0.ccs.ornl.gov
CRAY_PMI_POST_LINK_OPTS=-L/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/lib64
HOME=/ccs/home/roche
SHLVL=2
QT_SYSTEM_DIR=/usr/share/desktop-data
OSTYPE=linux
LESS_ADVANCED_PREPROCESSOR=no
PGI_PATH=/opt/pgi/12.1.0
ALTD_PATH=/sw/xk6/altd
LINKER_X86_64=/sw/xk6/altd/bin/ld
XCURSOR_THEME=DMZ
LS_OPTIONS=-N --color=none -T 0
CRAY_MPICH2_BASEDIR=/opt/cray/mpt/5.4.1/xt/gemini
CRAY_PMI_INCLUDE_OPTS=-I/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/include
PBS_O_HOST=jaguarpf-ext1.ccs.ornl.gov
CRAY_LLM_DIR=/opt/cray/llm/default
WINDOWMANAGER=/usr/bin/gnome
PRGENVMODULES=/opt/modules/3.2.6.6/init/.prgenvmodules:PrgEnv-cray:PrgEnv-gnu:
PrgEnv-intel:PrgEnv-pathscale:PrgEnv-pgi
ATP_MRNET_COMM_PATH=/opt/cray/atp/1.4.1/bin/atp_mrnet_commnnode_wrapper
PBS_VNODENUM=0
XTPE_INTERLAGOS_ENABLED=ON
LOGNAME=roche
MACHTYPE=x86_64-suse-linux
LESS=-M -I
G_FILENAME_ENCODING=@locale,UTF-8,ISO-8859-15,CP1252
CRAY_GNI_HEADERS_INCLUDE_OPTS=-I/opt/cray/gni-headers/2.1-1.0400.3906.5.1.gem/include
ALTD_SELECT_USERS=
CVS_RSH=ssh
DMAPP_ABORT_ON_ERROR=1
PBS_QUEUE=debug
SSH_CONNECTION=166.250.9.42 4993 160.91.205.198 22
XDG_DATA_DIRS=/usr/share:/etc/opt/kde3/share:/opt/kde3/share
TOOLMODULES=/opt/modules/3.2.6.6/init/.toolmodules:apprentice2:atp:craypat:
mrnet:stat:xt-crpat:xt-lgdb:xt-papi:xt-totalview:
/etc/opt/cray/modules/site_toolmodules
MODULESHOME=/opt/modules/3.2.6.6
LESSOPEN=lessopen.sh %s
PKG_CONFIG_PATH=/opt/cray/MySQL/5.0.64-1.0000.4667.20.1/lib64/pkgconfig:
/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/lib64/pkgconfig:
/opt/cray/gni-headers/2.1-1.0400.3906.5.1.gem/lib64/pkgconfig:
/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/lib64/pkgconfig:
/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/lib64/pkgconfig:
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/lib64/pkgconfig:  
/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/lib64/pkgconfig  
PBS_O_MAIL=/var/mail/roche  
LIBSCI_BASE_DIR=/opt/xt-libsci/11.0.04.4  
INFOPATH=/usr/local/info:/usr/share/info:/usr/info  
CRAY_MPICH2_ROOTDIR=/opt/cray/mpt/5.4.1/xt  
LIBSCI_VERSION=11.0.04.4  
PBS_NP=1  
CRAY_PRE_COMPILE_OPTS=-hnetwork=gemini  
CRAY_CPU_TARGET=interlagos  
PBS_NUM_PPN=1  
CRAY_UGNI_INCLUDE_OPTS=-I/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/include  
CRAY_XPMEM_INCLUDE_OPTS=-I/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/include  
SYSROOT_DIR=/opt/cray/xe-sysroot/4.0.30.securitypatch.20110928  
LESSCLOSE=lessclose.sh %s %s  
ATP_HOME=/opt/cray/atp/1.4.1  
PBS_NODEFILE=/var/spool/torque/aux//1060095.jaguarpf-sys0.ccs.ornl.gov  
G_BROKEN_Filenames=1  
CRAY_LD_LIBRARY_PATH=/opt/cray/mpt/5.4.1/xt/gemini/mpich2-pgi/109/lib:  
/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/lib64:  
/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/lib64:  
/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/lib64:  
/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/lib64:  
/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/lib64  
COLORTERM=1  
PBS_O_PATH=/opt/cray/papi/4.2.0/perf_events/no-cuda/bin:/sw/xk6/altd/bin:  
/sw/xk6/bin:/sw/xk6/lustredu/1.0/sles11.1_gnu4.3.4/lustredu/bin:  
/opt/cray/eslogin/eswrap/1.0.9/bin:/opt/cray/atp/1.4.1/bin:  
/opt/cray/xt-asyncpe/5.05/bin:/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/bin:  
/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/bin:  
/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/bin:  
/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/bin:  
/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/bin:  
/opt/pgi/12.1.0/linux86-64/12.1/bin:/opt/modules/3.2.6.6/bin:/usr/local/bin:  
/usr/bin:/bin:/usr/bin/X11:/usr/X11R6/bin:/usr/games:/opt/bin:  
/usr/lib/mit/bin:/usr/lib/mit/sbin:/opt/dell/srvadmin/bin:  
/opt/bin:/opt/public/bin  
OLDPWD=/ccs/home/roche  
_=~/usr/bin/env
```

Compilation

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
jaguarpf> cat cmpl.f-usr-krp
ftn -c f-usr-krp.f90 ;
ftn -c -Minfo=mp -mp f-usr-krp-omp.f90 ;
ftn -c f-usr-krp-simple.f90 ;
cc -c get-rnd.c ;
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt-init.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt-init-sum.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-init-sum.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-init.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-stat.c;
ftn -o xf-usr-krp-omp f-usr-krp-omp.o krp-rpt.o krp-rpt-init.o krp-stat.o krp-rpt-init-su
krp-init-sum.o krp-init.o get-rnd.o ${PAPI_POST_LINK_OPTS} -mp -lm;
ftn -o xf-usr-krp f-usr-krp.o krp-rpt.o krp-rpt-init.o krp-rpt-init-sum.o krp-init-sum.o
krp-init.o get-rnd.o ${PAPI_POST_LINK_OPTS} -lm;
ftn -o xf-usr-krp-smpl f-usr-krp-simple.o krp-rpt.o krp-rpt-init.o krp-rpt-init-sum.o \\
krp-init-sum.o krp-init.o get-rnd.o ${PAPI_POST_LINK_OPTS} -lm

jaguarpf> source cmpl.f-usr-krp
f_usr_krp_omp:
    93, Parallel region activated
    95, Parallel region terminated
   147, Parallel region activated
   148, Barrier
   152, Begin critical section (__cs_unspc)
   155, End critical section (__cs_unspc)
   162, Begin sections
   175, New section
   184, End sections
        Barrier
   188, Barrier
   190, Parallel loop activated with static block schedule
   193, Barrier
   196, Begin critical section (__cs_unspc)
   222, End critical section (__cs_unspc)
   225, Parallel region terminated
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
```

Run Script

```
jaguarpf> qsub -I -A csc091 -q debug -V -lsize=64,walltime=59:00 -lgres=widow2%widow3
qsub: waiting for job 1060095.jaguarpf-sys0.ccs.ornl.gov to start
qsub: job 1060095.jaguarpf-sys0.ccs.ornl.gov ready
jaguarpf> cd $PBS_O_WORKDIR
```

Run Examples: Scenario 1 detailed

```
jaguarpf> time aprun -n 1 ./xf-usr-krp
m 1 n
512 1024 512
m2 12 n2
512 256 512
nits
3
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop

[... enter work phases:] PAPI_TOT_INS : Tot[ 111952054 ] Hi[ 0 , 111952054 ] Lo[ 0 , 111952054 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_FP_INS : Tot[ 3670016 ] Hi[ 0 , 3670016 ] Lo[ 0 , 3670016 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_L2_DCM : Tot[ 104 ] Hi[ 0 , 104 ] Lo[ 0 , 104 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_real_cyc = [ 0 , 88371431 ] Lo[ 0 , 88371431 ]
[... enter work phases:] PAPI_real_usec = [ 0 , 40169 ] Lo[ 0 , 40169 ]
[... enter work phases:] PAPI_user_cyc = [ 0 , 88000000 ] Lo[ 0 , 88000000 ]
[... enter work phases:] PAPI_user_usec = [ 0 , 40000 ] Lo[ 0 , 40000 ]
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
.... entering phase 1 work          1
.... entering phase 2 work          1
.... entering phase 1 work          2
.... entering phase 2 work          2
.... entering phase 1 work          3
.... entering phase 2 work          3

[work phase 1] PAPI_TOT_INS : Tot[ 24996773834 ] Hi[ 0 , 24996773834 ] Lo[ 0 , 24996773834 ] SDev[ 0.000000 ]
[work phase 1] PAPI_FP_INS : Tot[ 6453460992 ] Hi[ 0 , 6453460992 ] Lo[ 0 , 6453460992 ] SDev[ 0.000000 ]
[work phase 1] PAPI_L2_DCM : Tot[ 386235329 ] Hi[ 0 , 386235329 ] Lo[ 0 , 386235329 ] SDev[ 0.000000 ]
[work phase 1] PAPI_real_cyc = [ 0 , 16035655174 ] Lo[ 0 , 16035655174 ]
[work phase 1] PAPI_real_usec = [ 0 , 7288934 ] Lo[ 0 , 7288934 ]
[work phase 1] PAPI_user_cyc = [ 0 , 16038000000 ] Lo[ 0 , 16038000000 ]
[work phase 1] PAPI_user_usec = [ 0 , 7290000 ] Lo[ 0 , 7290000 ]

[work phase 2] PAPI_TOT_INS : Tot[ 6272604919 ] Hi[ 0 , 6272604919 ] Lo[ 0 , 6272604919 ] SDev[ 0.000000 ]
[work phase 2] PAPI_FP_INS : Tot[ 1621622784 ] Hi[ 0 , 1621622784 ] Lo[ 0 , 1621622784 ] SDev[ 0.000000 ]
[work phase 2] PAPI_L2_DCM : Tot[ 775972 ] Hi[ 0 , 775972 ] Lo[ 0 , 775972 ] SDev[ 0.000000 ]
[work phase 2] PAPI_real_cyc = [ 0 , 2015479107 ] Lo[ 0 , 2015479107 ]
[work phase 2] PAPI_real_usec = [ 0 , 916127 ] Lo[ 0 , 916127 ]
[work phase 2] PAPI_user_cyc = [ 0 , 2024000000 ] Lo[ 0 , 2024000000 ]
[work phase 2] PAPI_user_usec = [ 0 , 920000 ] Lo[ 0 , 920000 ]

PREDICTION P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
== 6.4518881E+09

PREDICTION P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 1.6200499E+09

Application 382486 resources: utime ~25s, stime ~0s
real 0m53.236s
user 0m0.356s
sys 0m0.068s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 1 ./xf-usr-krp
m 1 n
512 1024 512
m2 12 n2
512 256 512
nits
1
Initialize PAPI Hardware Event Profilers
TotPEs()[16]
Mhz[2200.01]
nCPU-SMPnode()[8]
nSMPnodes()[2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop

[... enter work phases:] PAPI_TOT_INS : Tot[ 111952053 ] Hi[ 0 , 111952053 ] Lo[ 0 , 111952053 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_FP_INS : Tot[ 3670016 ] Hi[ 0 , 3670016 ] Lo[ 0 , 3670016 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_L2_DCM : Tot[ 107 ] Hi[ 0 , 107 ] Lo[ 0 , 107 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_real_cyc = [ 0 , 88169498 ] Lo[ 0 , 88169498 ]
[... enter work phases:] PAPI_real_usec = [ 0 , 40077 ] Lo[ 0 , 40077 ]
[... enter work phases:] PAPI_user_cyc = [ 0 , 88000000 ] Lo[ 0 , 88000000 ]
[... enter work phases:] PAPI_user_usec = [ 0 , 40000 ] Lo[ 0 , 40000 ]

.... entering phase 1 work          1
.... entering phase 2 work          1

[work phase 1] PAPI_TOT_INS : Tot[ 8332260001 ] Hi[ 0 , 8332260001 ] Lo[ 0 , 8332260001 ] SDev[ 0.000000 ]
[work phase 1] PAPI_FP_INS : Tot[ 2151153664 ] Hi[ 0 , 2151153664 ] Lo[ 0 , 2151153664 ] SDev[ 0.000000 ]
[work phase 1] PAPI_L2_DCM : Tot[ 135025009 ] Hi[ 0 , 135025009 ] Lo[ 0 , 135025009 ] SDev[ 0.000000 ]
[work phase 1] PAPI_real_cyc = [ 0 , 16075138196 ] Lo[ 0 , 16075138196 ]
[work phase 1] PAPI_real_usec = [ 0 , 7306881 ] Lo[ 0 , 7306881 ]
[work phase 1] PAPI_user_cyc = [ 0 , 16082000000 ] Lo[ 0 , 16082000000 ]
[work phase 1] PAPI_user_usec = [ 0 , 7310000 ] Lo[ 0 , 7310000 ]

[work phase 2] PAPI_TOT_INS : Tot[ 2090869333 ] Hi[ 0 , 2090869333 ] Lo[ 0 , 2090869333 ] SDev[ 0.000000 ]
[work phase 2] PAPI_FP_INS : Tot[ 540540928 ] Hi[ 0 , 540540928 ] Lo[ 0 , 540540928 ] SDev[ 0.000000 ]
[work phase 2] PAPI_L2_DCM : Tot[ 242759 ] Hi[ 0 , 242759 ] Lo[ 0 , 242759 ] SDev[ 0.000000 ]
[work phase 2] PAPI_real_cyc = [ 0 , 2016697862 ] Lo[ 0 , 2016697862 ]
[work phase 2] PAPI_real_usec = [ 0 , 916681 ] Lo[ 0 , 916681 ]
[work phase 2] PAPI_user_cyc = [ 0 , 2002000000 ] Lo[ 0 , 2002000000 ]
[work phase 2] PAPI_user_usec = [ 0 , 910000 ] Lo[ 0 , 910000 ]

PREDICTION P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
== 2.1506294E+09

PREDICTION P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 5.4001664E+08

Application 382494 resources: utime ~8s, stime ~0s
real 0m27.715s
user 0m0.348s
sys 0m0.092s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 3 ./xf-usr-krp
m 1 n
512 1024 512
m2 12 n2
512 256 512
nits
1
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop

[... enter work phases:] PAPI_TOT_INS : Tot[ 127056531676 ] Hi[ 2 , 64484226988 ] Lo[ 0 , 111956758 ] SDev[ 29879772315.699318 ]
[... enter work phases:] PAPI_FP_INS : Tot[ 11010048 ] Hi[ 0 , 3670016 ] Lo[ 0 , 3670016 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_L2_DCM : Tot[ 304 ] Hi[ 0 , 119 ] Lo[ 1 , 90 ] SDev[ 12.657892 ]
[... enter work phases:] PAPI_real_cyc = [ 1 , 28163695343 ] Lo[ 0 , 120697025 ]
[... enter work phases:] PAPI_real_usec = [ 1 , 12801680 ] Lo[ 0 , 54862 ]
[... enter work phases:] PAPI_user_cyc = [ 1 , 28160000000 ] Lo[ 0 , 110000000 ]
[... enter work phases:] PAPI_user_usec = [ 1 , 12800000 ] Lo[ 0 , 50000 ]

.... entering phase 1 work          1
.... entering phase 2 work          1

[work phase 1] PAPI_TOT_INS : Tot[ 24996776192 ] Hi[ 0 , 8332257838 ] Lo[ 2 , 8332257838 ] SDev[ 1077.818888 ]
[work phase 1] PAPI_FP_INS : Tot[ 6453460992 ] Hi[ 0 , 2151153664 ] Lo[ 0 , 2151153664 ] SDev[ 0.000000 ]
[work phase 1] PAPI_L2_DCM : Tot[ 379923845 ] Hi[ 1 , 135523528 ] Lo[ 2 , 108878838 ] SDev[ 12559944.395000 ]
[work phase 1] PAPI_real_cyc = [ 0 , 17750715032 ] Lo[ 2 , 15453405332 ]
[work phase 1] PAPI_real_usec = [ 0 , 8068507 ] Lo[ 2 , 7024275 ]
[work phase 1] PAPI_user_cyc = [ 0 , 17754000000 ] Lo[ 2 , 15444000000 ]
[work phase 1] PAPI_user_usec = [ 0 , 8070000 ] Lo[ 2 , 7020000 ]

[work phase 2] PAPI_TOT_INS : Tot[ 6272604642 ] Hi[ 0 , 2090869662 ] Lo[ 2 , 2090867453 ] SDev[ 1024.336208 ]
[work phase 2] PAPI_FP_INS : Tot[ 1621622784 ] Hi[ 0 , 540540928 ] Lo[ 0 , 540540928 ] SDev[ 0.000000 ]
[work phase 2] PAPI_L2_DCM : Tot[ 987665 ] Hi[ 1 , 371050 ] Lo[ 2 , 246467 ] SDev[ 58517.544619 ]
[work phase 2] PAPI_real_cyc = [ 0 , 2314441900 ] Lo[ 2 , 2043638756 ]
[work phase 2] PAPI_real_usec = [ 0 , 1052019 ] Lo[ 2 , 928927 ]
[work phase 2] PAPI_user_cyc = [ 0 , 231000000 ] Lo[ 2 , 2046000000 ]
[work phase 2] PAPI_user_usec = [ 0 , 1050000 ] Lo[ 2 , 930000 ]

PREDICTION P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n.n)
== 6.4518881E+09

PREDICTION P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 1.6200499E+09

Application 382502 resources: utime ~53s, stime ~1s
real 0m22.873s
user 0m0.356s
sys 0m0.080s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 3 ./xf-usr-krp
m 1 n
512 1024 512
m2 12 n2
512 256 512
nits
10
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
model number[1]

.... initialize the arrays
.... entering loop

[... enter work phases:] PAPI_TOT_INS : Tot[ 238512518441 ] Hi[ 2 , 121011741728 ] Lo[ 0 , 111956755 ] SDev[ 56158254772.071823 ]
[... enter work phases:] PAPI_FP_INS : Tot[ 11010048 ] Hi[ 0 , 3670016 ] Lo[ 0 , 3670016 ] SDev[ 0.000000 ]
[... enter work phases:] PAPI_L2_DCM : Tot[ 340 ] Hi[ 0 , 140 ] Lo[ 2 , 87 ] SDev[ 21.638443 ]
[... enter work phases:] PAPI_real_cyc = [ 1 , 52841889661 ] Lo[ 0 , 120585652 ]
[... enter work phases:] PAPI_real_usec = [ 1 , 24019041 ] Lo[ 0 , 54811 ]
[... enter work phases:] PAPI_user_cyc = [ 1 , 52844000000 ] Lo[ 0 , 88000000 ]
[... enter work phases:] PAPI_user_usec = [ 1 , 24020000 ] Lo[ 0 , 40000 ]

.... entering phase 1 work      1
.... entering phase 2 work      1
.... entering phase 1 work      2
.... entering phase 2 work      2
.... entering phase 1 work      3
.... entering phase 2 work      3
.... entering phase 1 work      4
.... entering phase 2 work      4
.... entering phase 1 work      5
.... entering phase 2 work      5
.... entering phase 1 work      6
.... entering phase 2 work      6
.... entering phase 1 work      7
.... entering phase 2 work      7
.... entering phase 1 work      8
.... entering phase 2 work      8
.... entering phase 1 work      9
.... entering phase 2 work      9
.... entering phase 1 work     10
.... entering phase 2 work     10

[work phase 1] PAPI_TOT_INS : Tot[ 249967678724 ] Hi[ 0 , 83322573991 ] Lo[ 2 , 83322552191 ] SDev[ 10194.894158 ]
[work phase 1] PAPI_FP_INS : Tot[ 64534609920 ] Hi[ 0 , 21511536640 ] Lo[ 0 , 21511536640 ] SDev[ 0.000000 ]
[work phase 1] PAPI_L2_DCM : Tot[ 3100363431 ] Hi[ 1 , 1144780335 ] Lo[ 2 , 821106730 ] SDev[ 150211444.717705 ]
[work phase 1] PAPI_real_cyc = [ 0 , 16120439368 ] Lo[ 2 , 14998202361 ]
[work phase 1] PAPI_real_usec = [ 0 , 7327472 ] Lo[ 2 , 6817365 ]
[work phase 1] PAPI_user_cyc = [ 0 , 16104000000 ] Lo[ 2 , 15004000000 ]
[work phase 1] PAPI_user_usec = [ 0 , 7320000 ] Lo[ 2 , 6820000 ]

[work phase 2] PAPI_TOT_INS : Tot[ 62726003055 ] Hi[ 0 , 20908682825 ] Lo[ 2 , 20908658772 ] SDev[ 10761.609204 ]
[work phase 2] PAPI_FP_INS : Tot[ 16216227840 ] Hi[ 0 , 5405409280 ] Lo[ 0 , 5405409280 ] SDev[ 0.000000 ]
[work phase 2] PAPI_L2_DCM : Tot[ 10650240 ] Hi[ 0 , 3978803 ] Lo[ 2 , 2743265 ] SDev[ 570878.683965 ]
[work phase 2] PAPI_real_cyc = [ 0 , 2310211843 ] Lo[ 2 , 2041144172 ]
[work phase 2] PAPI_real_usec = [ 0 , 1050097 ] Lo[ 2 , 927793 ]
[work phase 2] PAPI_user_cyc = [ 0 , 2310000000 ] Lo[ 2 , 2046000000 ]
[work phase 2] PAPI_user_usec = [ 0 , 1050000 ] Lo[ 2 , 930000 ]

PREDICTION P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
== 6.4518881E+10

PREDICTION P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 1.6200499E+10

Application 382508 resources: utime ~311s, stime ~1s
real 1m52.841s
user 0m0.372s
sys 0m0.060s
```

Run Examples: Scenario 1 simple

```
roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 1 ./xf-usr-krp-smpl
m l n
512 1024 512
m2 l2 n2
512 256 512
nits
1
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
model number[1]

.... initialize the arrays
.... entering loop
.... entering phase 1 work      1
.... entering phase 2 work      1

[... exiting] PAPI_TOT_INS : Tot[ 10535081928 ] Hi[ 0 , 10535081928 ] Lo[ 0 , 10535081928 ] SDev[ 0.000000 ]
[... exiting] PAPI_FP_INS : Tot[ 2695364608 ] Hi[ 0 , 2695364608 ] Lo[ 0 , 2695364608 ] SDev[ 0.000000 ]
[... exiting] PAPI_L2_DCM : Tot[ 135260353 ] Hi[ 0 , 135260353 ] Lo[ 0 , 135260353 ] SDev[ 0.000000 ]
PAPI_real_cyc = [ 0 , 18236155477 ] Lo[ 0 , 18236155477 ]
PAPI_real_usec = [ 0 , 8289162 ] Lo[ 0 , 8289162 ]
PAPI_user_cyc = [ 0 , 18238000000 ] Lo[ 0 , 18238000000 ]
PAPI_user_usec = [ 0 , 8290000 ] Lo[ 0 , 8290000 ]

PREDICTION (just the loop) P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
== 2.1506294E+09

PREDICTION (just the loop) P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 5.4001664E+08

Application 382515 resources: utime ~8s, stime ~0s
real 0m26.254s
user 0m0.376s
sys 0m0.056s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 1 ./xf-usr-krp-smpl
m 1 n
512 1024 512
m2 12 n2
512 256 512
nits
3
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop
.... entering phase 1 work      1
.... entering phase 2 work      1
.... entering phase 1 work      2
.... entering phase 2 work      2
.... entering phase 1 work      3
.... entering phase 2 work      3

[... exiting] PAPI_TOT_INS : Tot[ 31381331272 ] Hi[ 0 , 31381331272 ] Lo[ 0 , 31381331272 ] SDev[ 0.000000 ]
[... exiting] PAPI_FP_INS : Tot[ 8078753792 ] Hi[ 0 , 8078753792 ] Lo[ 0 , 8078753792 ] SDev[ 0.000000 ]
[... exiting] PAPI_L2_DCM : Tot[ 404285556 ] Hi[ 0 , 404285556 ] Lo[ 0 , 404285556 ] SDev[ 0.000000 ]
PAPI_real_cyc = [ 0 , 54115267722 ] Lo[ 0 , 54115267722 ]
PAPI_real_usec = [ 0 , 24597849 ] Lo[ 0 , 24597849 ]
PAPI_user_cyc = [ 0 , 54120000000 ] Lo[ 0 , 54120000000 ]
PAPI_user_usec = [ 0 , 24600000 ] Lo[ 0 , 24600000 ]

PREDICTION (just the loop) P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
== 6.4518881E+09

PREDICTION (just the loop) P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 1.6200499E+09

Application 382516 resources: utime ~25s, stime ~0s
real 0m37.055s
user 0m0.364s
sys 0m0.060s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 1 ./xf-usr-krp-smpl
m 1 n
512 1024 512
m2 12 n2
512 256 512
nits
3
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop
.... entering phase 1 work      1
.... entering phase 2 work      1
.... entering phase 1 work      2
.... entering phase 2 work      2
.... entering phase 1 work      3
.... entering phase 2 work      3

[... exiting] PAPI_TOT_INS : Tot[ 31381331261 ] Hi[ 0 , 31381331261 ] Lo[ 0 , 31381331261 ] SDev[ 0.000000 ]
[... exiting] PAPI_FP_INS : Tot[ 8078753792 ] Hi[ 0 , 8078753792 ] Lo[ 0 , 8078753792 ] SDev[ 0.000000 ]
[... exiting] PAPI_L2_DCM : Tot[ 401982054 ] Hi[ 0 , 401982054 ] Lo[ 0 , 401982054 ] SDev[ 0.000000 ]
PAPI_real_cyc = [ 0 , 54031075989 ] Lo[ 0 , 54031075989 ]
PAPI_real_usec = [ 0 , 24559580 ] Lo[ 0 , 24559580 ]
PAPI_user_cyc = [ 0 , 54032000000 ] Lo[ 0 , 54032000000 ]
PAPI_user_usec = [ 0 , 24560000 ] Lo[ 0 , 24560000 ]

PREDICTION (just the loop) P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
== 6.4518881E+09

PREDICTION (just the loop) P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 1.6200499E+09

Application 382530 resources: utime ~25s, stime ~0s
real 0m39.397s
user 0m0.356s
sys 0m0.088s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 3 ./xf-usr-krp-smpl
m l n
512 1024 512
m2 l2 n2
512 256 512
nits
1
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop
.... entering phase 1 work      1
.... entering phase 2 work      1

[... exiting] PAPI_TOT_INS : Tot[ 130589848655 ] Hi[ 1 , 60132731484 ] Lo[ 0 , 10535087165 ] SDev[ 23331049507.962719 ]
[... exiting] PAPI_FP_INS : Tot[ 8086093824 ] Hi[ 0 , 2695364608 ] Lo[ 0 , 2695364608 ] SDev[ 0.000000 ]
[... exiting] PAPI_L2_DCM : Tot[ 396549043 ] Hi[ 0 , 134611461 ] Lo[ 2 , 127479243 ] SDev[ 3326655.994031 ]
PAPI_real_cyc = [ 1 , 41951970677 ] Lo[ 0 , 20127493966 ]
PAPI_real_usec = [ 1 , 19069078 ] Lo[ 0 , 9148861 ]
PAPI_user_cyc = [ 1 , 41954000000 ] Lo[ 0 , 20130000000 ]
PAPI_user_usec = [ 1 , 19070000 ] Lo[ 0 , 9150000 ]

PREDICTION (just the loop) P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
== 6.4518881E+09

PREDICTION (just the loop) P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 1.6200499E+09

Application 382548 resources: utime ~47s, stime ~0s
real 0m19.791s
user 0m0.376s
sys 0m0.052s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 3 ./xf-usr-krp-smpl
m l n
512 1024 512
m2 l2 n2
512 256 512
nits
10
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop
.... entering phase 1 work      1
.... entering phase 2 work      1
.... entering phase 1 work      2
.... entering phase 2 work      2
.... entering phase 1 work      3
.... entering phase 2 work      3
.... entering phase 1 work      4
.... entering phase 2 work      4
.... entering phase 1 work      5
.... entering phase 2 work      5
.... entering phase 1 work      6
.... entering phase 2 work      6
.... entering phase 1 work      7
.... entering phase 2 work      7
.... entering phase 1 work      8
.... entering phase 2 work      8
.... entering phase 1 work      9
.... entering phase 2 work      9
.... entering phase 1 work     10
.... entering phase 2 work     10

[... exiting] PAPI_TOT_INS : Tot[ 415034272684 ] Hi[ 1 , 155401916435 ] Lo[ 0 , 104343214191 ] SDev[ 24042765969.393246 ]
[... exiting] PAPI_FP_INS : Tot[ 80761847808 ] Hi[ 0 , 26920615936 ] Lo[ 0 , 26920615936 ] SDev[ 0.000000 ]
[... exiting] PAPI_L2_DCM : Tot[ 3430369856 ] Hi[ 1 , 1356248231 ] Lo[ 2 , 719814697 ] SDev[ 299561123.990196 ]
PAPI_real_cyc = [ 1 , 222268503814 ] Lo[ 2 , 188025211747 ]
PAPI_real_usec = [ 1 , 101031138 ] Lo[ 2 , 85466005 ]
PAPI_user_cyc = [ 1 , 222266000000 ] Lo[ 2 , 188034000000 ]
PAPI_user_usec = [ 1 , 101030000 ] Lo[ 2 , 85470000 ]

PREDICTION (just the loop) P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
==   6.4518881E+10

PREDICTION (just the loop) P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
==   1.6200499E+10

Application 382565 resources: utime ~293s, stime ~0s
real 0m53.245s
user 0m0.352s
sys 0m0.088s

roche@jaguarpf-login1:/tmp/work/roche/kr-tsts/mm-study/f-ex> time aprun -n 3 ./xf-usr-krp-smpl
 m l n
512 1024 512
m2 12 n2
512 256 512
nits
10
Initialize PAPI Hardware Event Profilers
TotPEs() [16]
Mhz[2200.01]
nCPU-SMPnode() [8]
nSMPnodes() [2]
vendor string cpu[AuthenticAMD]
model string cpu[AMD Opteron(TM) Processor 6274 ]
model number[1]

.... initialize the arrays
.... entering loop
.... entering phase 1 work      1
.... entering phase 2 work      1
.... entering phase 1 work      2
.... entering phase 2 work      2
.... entering phase 1 work      3
.... entering phase 2 work      3
.... entering phase 1 work      4
.... entering phase 2 work      4
.... entering phase 1 work      5
.... entering phase 2 work      5
.... entering phase 1 work      6
.... entering phase 2 work      6
.... entering phase 1 work      7
.... entering phase 2 work      7
.... entering phase 1 work      8
.... entering phase 2 work      8
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```

.... entering phase 1 work          9
.... entering phase 2 work          9
.... entering phase 1 work          10
.... entering phase 2 work          10

[... exiting] PAPI_TOT_INS : Tot[ 407899090492 ] Hi[ 1 , 151963305730 ] Lo[ 0 , 104343214405 ] SDev[ 22361455379.978519 ]
[... exiting] PAPI_FP_INS : Tot[ 80761847808 ] Hi[ 0 , 26920615936 ] Lo[ 0 , 26920615936 ] SDev[ 0.000000 ]
[... exiting] PAPI_L2_DCM : Tot[ 3684914784 ] Hi[ 1 , 1341127942 ] Lo[ 2 , 1004852784 ] SDev[ 158007064.769056 ]
PAPI_real_cyc = [ 1 , 221732636686 ] Lo[ 2 , 193362804910 ]
PAPI_real_usec = [ 1 , 100787562 ] Lo[ 2 , 87892184 ]
PAPI_user_cyc = [ 1 , 221760000000 ] Lo[ 2 , 193380000000 ]
PAPI_user_usec = [ 1 , 100800000 ] Lo[ 2 , 87900000 ]

PREDICTION (just the loop) P1( TOTAL FP_OPS ) = PEs * nits * (8.m.n.l + 12.m.n)
== 6.4518881E+10

PREDICTION (just the loop) P2( FP_OPS ) = PEs * nits * (8.mm.nn.ll + 12.mm.nn)
== 1.6200499E+10

Application 382637 resources: utime ~292s, stime ~1s
real 1m41.553s
user 0m0.348s
sys 0m0.076s

```

8 Multiple subgroups with parallel work phase in Loop: calculate $C \leftarrow \alpha A B + \beta C$

Input: $np, ngrp, \{m, l, n, p, q, nb, nits\}_{igrp} \forall igrp \in [0, ngrp - 1]$

get MPI process ID $iam \in [0, np - 1]$;

for $igrp = 0 \rightarrow ngrp - 1$ **do**

 select $(pq)_{igrp}$ ranks of MPI processes for group $igrp$;

 form subgroup $igrp, igrp(iam) \cap igrp'(iam) = \emptyset$; \triangleright no common processes between groups

if $iam \in ranks$ **then**

$iam \rightarrow pid_{igrp}$; \triangleright assign MPI processes to group rank in $igrp$

$pid_{igrp} \rightarrow (ip, iq)_{igrp} \in ([0, p - 1], [0, q - 1])_{igrp}$; \triangleright rank in virtual process grid

 initialize α, β ;

 initialize $f(A(m_{igrp}, l_{igrp}), B(l_{igrp}, n_{igrp}), C(m_{igrp}, n_{igrp}), ip_{igrd}, iq_{igrd}, nb_{igrp}, p_{igrd}, q_{igrd})$

$\rightarrow A(m_{ip_{igrp}}, l_{iq_{igrp}}), B(l_{ip_{igrp}}, n_{iq_{igrp}}), C(m_{ip_{igrp}}, n_{iq_{igrp}})$; \triangleright distributed block cyclic arrays

end if

end for

for $igrp = 0 \rightarrow ngrp - 1$ **do**

if $iam \in igrp$ **then**

for $i = 0 \rightarrow nits_{igrp}$ **do** \triangleright ie $op(A)_{igrp} := A(m_{ip_{igrp}}, l_{iq_{igrp}})$

 parallel $zgemm(pid_{igrp}, m_{igrp}, l_{igrp}, n_{igrp}, op(A)_{igrp}, op(B)_{igrp}, op(C)_{igrp}, \alpha, \beta)$

end for

end if

Compilation

```

jaguarpf> cat cmpl.c-usr-krp
cc -c c-usr-krp.c ;
cc -c -DKRP -Minfo -mp mm-omp.c ;

```

9 Multi-threaded work phase: calculate $C \leftarrow \alpha AB + \beta C$

Input: $np, d, m, l, n, chnk$ ▷ typically $chnk * d = k * l$ and k integral
 get MPI process ID $ip \in [0, np - 1]$;
 ip creates $nt = d - 1$ threads of execution, $tid \in [0, nt - 1]$; ▷ $tid = 0$ shares ip 's cpu
 initialize α, β ;
 threaded initialization $f(A(m, l), B(l, n), C(m, n), chnk, tid) \rightarrow A(ml_{tid}), B(ln_{tid}), C(mn_{tid})$; ▷ block cyclic chunking schedule
 threaded $zgemm(chnk, tid, m, l, n, op(A)_{tid}, op(B)_{tid}, op(C)_{tid}, \alpha, \beta)$; ▷ ie
 $op(A)_{tid} := A(ml_{tid})$, $ml_{tid} \sim m * l / nt$, block cyclic chunking = 0

```

cc -c -DKRP c-usr-krp-smpl.c ;
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt-init.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt-init-sum.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-init-sum.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-init.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-stat.c;
cc -o xc-usr-nokrp c-usr-krp.o krp-rpt.o krp-rpt-init.o krp-rpt-init-sum.o krp-init-sum.o \\
krp-init.o ${PAPI_POST_LINK_OPTS} -lm;
cc -c -DKRP c-usr-krp.c ;
cc -o xc-usr-krp c-usr-krp.o krp-rpt.o krp-rpt-init.o krp-rpt-init-sum.o krp-init-sum.o \\
krp-init.o ${PAPI_POST_LINK_OPTS} -lm;
cc -o xc-usr-krp-smpl c-usr-krp-smpl.o krp-rpt.o krp-rpt-init.o krp-rpt-init-sum.o krp-init-sum.o
krp-init.o ${PAPI_POST_LINK_OPTS} -lm;
cc -o xmm-omp mm-omp.o krp-rpt.o krp-stat.o krp-rpt-init.o krp-rpt-init-sum.o krp-init-sum.o \\
krp-init.o ${PAPI_POST_LINK_OPTS} -mp -lm ;

jaguarpf> source cmpl.c-usr-krp
main:
  111, Parallel region activated
  133, Parallel loop activated with static block-cyclic schedule
  137, Barrier
  138, Parallel loop activated with static block-cyclic schedule
  142, Barrier
  143, Parallel loop activated with static block-cyclic schedule
  148, Barrier
  151, Begin critical section (__cs_unspc)
  160, End critical section (__cs_unspc)
  162, Parallel region terminated
  163, Parallel region activated
  190, Barrier

```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
198, Parallel loop activated with static block-cyclic schedule
220, Barrier
224, Begin critical section (__cs_unspc)
233, End critical section (__cs_unspc)
237, Parallel region terminated
238, Parallel region activated
253, Parallel region terminated
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
PGC-W-0043-Redefinition of symbol,
caddr_t (/opt/cray/papi/4.2.0/perf_events/no-cuda/include/papi.h: 585)
PGC/x86-64 Linux 12.1-0: compilation completed with warnings
```

Run Examples: Scenario 2 -detailed

```
roche@jaguarpf-login5:/tmp/work/roche/kr-tsts/mm-study/c-api> time aprun -n 57 ./xc-usr-krp 3
grp[ 0]: m l n p q nb nits
4096 2048 4096 4 4 40 10
pes[0] 16
grp[ 1]: m l n p q nb nits
4096 4096 4096 4 4 40 3
pes[1] 32
grp[ 2]: m l n p q nb nits
8192 8192 8192 5 5 40 1
pes[2] 57

[GRP[0] Init] PAPI_TOT_INS : Tot[ 3946417338 ] Hi[ 0 , 263143852 ] Lo[ 15 , 232070505 ] SDev[ 7320288.283303 ]
[GRP[0] Init] PAPI_FP_INS : Tot[ 134217856 ] Hi[ 0 , 8652808 ] Lo[ 15 , 7904008 ] SDev[ 217992.868874 ]
[GRP[0] Init] PAPI_L2_DCM : Tot[ 2859 ] Hi[ 0 , 305 ] Lo[ 11 , 128 ] SDev[ 45.103657 ]
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
[GRP[0] Init] PAPI_real_cyc = [ 0 , 696775855 ] Lo[ 14 , 292931533 ]
[GRP[0] Init] PAPI_real_usec = [ 0 , 316716 ] Lo[ 14 , 133150 ]
[GRP[0] Init] PAPI_user_cyc = [ 0 , 638000000 ] Lo[ 2 , 286000000 ]
[GRP[0] Init] PAPI_user_usec = [ 0 , 290000 ] Lo[ 2 , 130000 ]

[GRP[1] Init] PAPI_TOT_INS : Tot[ 5911477646 ] Hi[ 0 , 393941156 ] Lo[ 15 , 351811438 ] SDev[ 10190940.036088 ]
[GRP[1] Init] PAPI_FP_INS : Tot[ 201326720 ] Hi[ 0 , 12979208 ] Lo[ 15 , 12000008 ] SDev[ 294932.249305 ]
[GRP[1] Init] PAPI_L2_DCM : Tot[ 2918 ] Hi[ 0 , 256 ] Lo[ 9 , 117 ] SDev[ 46.420463 ]
[GRP[1] Init] PAPI_real_cyc = [ 0 , 971889263 ] Lo[ 14 , 436877098 ]
[GRP[1] Init] PAPI_real_usec = [ 0 , 441768 ] Lo[ 14 , 198580 ]
[GRP[1] Init] PAPI_user_cyc = [ 0 , 968000000 ] Lo[ 10 , 418000000 ]
[GRP[1] Init] PAPI_user_usec = [ 0 , 440000 ] Lo[ 10 , 190000 ]

[WORLD: past Init] PAPI_TOT_INS : Tot[ 114331433362 ] Hi[ 6 , 4230765650 ] Lo[ 32 , 13187 ] SDev[ 1828835525.110472 ]
[WORLD: past Init] PAPI_FP_INS : Tot[ 0 ] Hi[ 0 , 0 ] Lo[ 0 , 0 ] SDev[ 0.000000 ]
[WORLD: past Init] PAPI_L2_DCM : Tot[ 5119 ] Hi[ 14 , 309 ] Lo[ 50 , 21 ] SDev[ 68.023587 ]
[WORLD: past Init] PAPI_real_cyc = [ 56 , 4068198937 ] Lo[ 23 , 4061701635 ]
[WORLD: past Init] PAPI_real_usec = [ 56 , 1849181 ] Lo[ 22 , 1846228 ]
[WORLD: past Init] PAPI_user_cyc = [ 7 , 4092000000 ] Lo[ 31 , 3894000000 ]
[WORLD: past Init] PAPI_user_usec = [ 7 , 1860000 ] Lo[ 31 , 1770000 ]

[GRP[2] Init] PAPI_TOT_INS : Tot[ 23642932421 ] Hi[ 16 , 978093158 ] Lo[ 24 , 938685259 ] SDev[ 9695977.494729 ]
[GRP[2] Init] PAPI_FP_INS : Tot[ 805306568 ] Hi[ 0 , 32275208 ] Lo[ 24 , 31961096 ] SDev[ 88974.708876 ]
[GRP[2] Init] PAPI_L2_DCM : Tot[ 4088 ] Hi[ 8 , 295 ] Lo[ 5 , 93 ] SDev[ 63.705020 ]
[GRP[2] Init] PAPI_real_cyc = [ 0 , 2387915611 ] Lo[ 24 , 794228831 ]
[GRP[2] Init] PAPI_real_usec = [ 0 , 1085416 ] Lo[ 24 , 361013 ]
[GRP[2] Init] PAPI_user_cyc = [ 0 , 2376000000 ] Lo[ 24 , 770000000 ]
[GRP[2] Init] PAPI_user_usec = [ 0 , 1080000 ] Lo[ 24 , 350000 ]

[Work GRP[1]] PAPI_TOT_INS : Tot[ 926228706335 ] Hi[ 5 , 58222795460 ] Lo[ 15 , 57442070885 ] SDev[ 192571811.480662 ]
[Work GRP[1]] PAPI_FP_INS : Tot[ 1714044273504 ] Hi[ 0 , 110501664054 ] Lo[ 15 , 10216500054 ] SDev[ 2510979437.518904 ]
[Work GRP[1]] PAPI_L2_DCM : Tot[ 732393724 ] Hi[ 0 , 49200195 ] Lo[ 15 , 43152871 ] SDev[ 1541264.545592 ]
[Work GRP[1]] PAPI_real_cyc = [ 5 , 41554464794 ] Lo[ 15 , 41542898882 ]
[Work GRP[1]] PAPI_real_usec = [ 5 , 18888393 ] Lo[ 15 , 188883136 ]
[Work GRP[1]] PAPI_user_cyc = [ 3 , 41580000000 ] Lo[ 11 , 41404000000 ]
[Work GRP[1]] PAPI_user_usec = [ 3 , 18900000 ] Lo[ 11 , 18820000 ]

[Work GRP[2]] PAPI_TOT_INS : Tot[ 2468935609381 ] Hi[ 24 , 145468310497 ] Lo[ 22 , 96357095777 ] SDev[ 9536814268.777519 ]
[Work GRP[2]] PAPI_FP_INS : Tot[ 4570314965442 ] Hi[ 0 , 183169828818 ] Lo[ 24 , 181387164690 ] SDev[ 504953716.545733 ]
[Work GRP[2]] PAPI_L2_DCM : Tot[ 1686575355 ] Hi[ 6 , 71797348 ] Lo[ 24 , 37968117 ] SDev[ 6477208.714671 ]
[Work GRP[2]] PAPI_real_cyc = [ 18 , 68481265104 ] Lo[ 24 , 68371168852 ]
[Work GRP[2]] PAPI_real_usec = [ 18 , 31127848 ] Lo[ 24 , 31077804 ]
[Work GRP[2]] PAPI_user_cyc = [ 6 , 68508000000 ] Lo[ 15 , 68244000000 ]
[Work GRP[2]] PAPI_user_usec = [ 6 , 31140000 ] Lo[ 15 , 31020000 ]

[Work GRP[0]] PAPI_TOT_INS : Tot[ 1533124302178 ] Hi[ 1 , 96218834762 ] Lo[ 15 , 95205048552 ] SDev[ 261654335.452309 ]
[Work GRP[0]] PAPI_FP_INS : Tot[ 2857327659840 ] Hi[ 0 , 184207296180 ] Lo[ 15 , 17031000180 ] SDev[ 4185825948.258646 ]
[Work GRP[0]] PAPI_L2_DCM : Tot[ 1229089599 ] Hi[ 0 , 81847829 ] Lo[ 15 , 71791388 ] SDev[ 2557219.840526 ]
[Work GRP[0]] PAPI_real_cyc = [ 5 , 69995008435 ] Lo[ 15 , 69984211098 ]
[Work GRP[0]] PAPI_real_usec = [ 5 , 31815913 ] Lo[ 15 , 31811005 ]
[Work GRP[0]] PAPI_user_cyc = [ 1 , 70004000000 ] Lo[ 15 , 69762000000 ]
[Work GRP[0]] PAPI_user_usec = [ 1 , 31820000 ] Lo[ 15 , 31710000 ]

[WORLD: past loop] PAPI_TOT_INS : Tot[ 729215927863 ] Hi[ 30 , 38662640360 ] Lo[ 48 , 1833076769 ] SDev[ 16023507576.170149 ]
[WORLD: past loop] PAPI_FP_INS : Tot[ 0 ] Hi[ 0 , 0 ] Lo[ 0 , 0 ] SDev[ 0.000000 ]
[WORLD: past loop] PAPI_L2_DCM : Tot[ 7988 ] Hi[ 26 , 732 ] Lo[ 50 , 44 ] SDev[ 127.092374 ]
[WORLD: past loop] PAPI_real_cyc = [ 36 , 69998189950 ] Lo[ 21 , 69997882976 ]
[WORLD: past loop] PAPI_real_usec = [ 36 , 31817359 ] Lo[ 21 , 31817221 ]
[WORLD: past loop] PAPI_user_cyc = [ 6 , 70026000000 ] Lo[ 47 , 69762000000 ]
[WORLD: past loop] PAPI_user_usec = [ 6 , 31830000 ] Lo[ 47 , 31710000 ]
THY: subgroup[0] [4096,2048,4096] 10 times = FP[ 2750792335360.000000 ]
THY: subgroup[1] [4096,4096,4096] 3 times = FP[ 1649871421440.000000 ]
THY: subgroup[2] [8192,8192,8192] 1 times = FP[ 4398851817472.000000 ]
Total FP[ 8799515574272.000000 ]
Application 388125 resources: utime ~4688s, stime ~28s

real 1m26.383s
user 0m0.400s
sys 0m0.048s
roche@jaguarpf-login5:/tmp/work/roche/kr-tsts/mm-study/c-api>
```

Run Examples: Scenario 2 -simple

```
roche@jaguarpf-login5:/tmp/work/roche/kr-tsts/mm-study/c-api> time aprun -n 57 ./xc-usr-krp-smpl 3
grp[ 0]: m l n p q nb nits
4096 2048 4096 4 4 40 10
pes[0] 16
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
grp[ 1]: m l n p q nb nits
4096 4096 4096 4 4 40 3
pes[1] 32
grp[ 2]: m l n p q nb nits
8192 8192 8192 5 5 40 1
pes[2] 57

[WORLD: past loop] PAPI_TOT_INS : Tot[ 5739835192277 ] Hi[ 56 , 151744440801 ] Lo[ 16 , 96288598487 ] SDev[ 7024987811.039442 ]
[WORLD: past loop] PAPI_FP_INS : Tot[ 9142827749930 ] Hi[ 0 , 184215948988 ] Lo[ 31 , 102177000062 ] SDev[ 33419750547.725780 ]
[WORLD: past loop] PAPI_L2_DCM : Tot[ 3660573793 ] Hi[ 0 , 81637304 ] Lo[ 56 , 38148172 ] SDev[ 12762650.278293 ]
[WORLD: past loop] PAPI_real_cyc = [ 56 , 73977485163 ] Lo[ 19 , 73971127627 ]
[WORLD: past loop] PAPI_real_usec = [ 56 , 33626130 ] Lo[ 19 , 33623240 ]
[WORLD: past loop] PAPI_user_cyc = [ 1 , 73986000000 ] Lo[ 47 , 73744000000 ]
[WORLD: past loop] PAPI_user_usec = [ 1 , 33630000 ] Lo[ 47 , 33520000 ]
THY: subgroup[0] [4096,4096,4096] 10 times = FP[ 2750792335360.000000 ]
THY: subgroup[1] [4096,4096,4096] 3 times = FP[ 1649871421440.000000 ]
THY: subgroup[2] [8192,8192,8192] 1 times = FP[ 4398851817472.000000 ]
Total FP[ 8799515574272.000000 ]
Application 388135 resources: utime ~3516s, stime ~24s

real 1m5.511s
user 0m0.388s
sys 0m0.060s
roche@jaguarpf-login5:/tmp/work/roche/kr-tsts/mm-study/c-api>
```

Run Examples: Scenario 3

```
jaguarpf> export OMP_NUM_THREADS=2
jaguar> time aprun -n 2 -d 2 ./xzgemm-omp-krp 1024 1024 1024 512
[0][1] init[1024,1024,1024] FP[6291456] INS[224205574] L2DCM[49] USEC[1041729]
[0][0] init[1024,1024,1024] FP[6291456] INS[208882586] L2DCM[368] USEC[1041957]
... tid 0 starting matrix multiply
... tid 1 starting matrix multiply

[1][1] init[1024,1024,1024] FP[6291456] INS[207359185] L2DCM[1346] USEC[1082596]
[1][0] init[1024,1024,1024] FP[6291456] INS[250171551] L2DCM[85] USEC[1082814]
... tid 0 starting matrix multiply
... tid 1 starting matrix multiply

[0] mm[1024,1024,1024] FP[4302307328] INS[15051275232] L2DCM[540373568] USEC[20103149]
[0][0] mm[1024,1024,1024] FP[4302307328] INS[15305780632] L2DCM[540484806] USEC[20103253]

[1] mm[1024,1024,1024] FP[4302307328] INS[15365298220] L2DCM[540544637] USEC[20919046]
[1][0] mm[1024,1024,1024] FP[4302307328] INS[15051483168] L2DCM[540233428] USEC[20919151]

[ PAPI_TOT_INS ] : Tot[ 61664456148 ] Hi[ 1 , 30874312124 ] Lo[ 0 , 30790144024 ] SDev[ 42084050.000000 ]
[ PAPI_FP_INS ] : Tot[ 17234395136 ] Hi[ 0 , 8617197568 ] Lo[ 0 , 8617197568 ] SDev[ 0.000000 ]
[ PAPI_L2_DCM ] : Tot[ 2161638287 ] Hi[ 0 , 1080858791 ] Lo[ 1 , 1080779496 ] SDev[ 39647.500000 ]
[Real usec] : Tot[ 22001965 ] Hi[ 1 , 22001965 ] Lo[ 0 , 21145210 ] SDev[ 428377.500000 ]

THY: [1024,1024,1024] FP[ 17205035008 ]

Application 372241 resources: utime ~86s, stime ~2s
real    0m24.011s
user    0m0.336s
sys     0m0.060s

jaguarpf> export OMP_NUM_THREADS=4
jaguarpf> time aprun -n 2 -d 4 ./xzgemm-omp-krp 1024 1024 1024 256
[0][1] init[1024,1024,1024] FP[3145728] INS[126814035] L2DCM[2107375] USEC[2421820]
[0][3] init[1024,1024,1024] FP[3145728] INS[107498702] L2DCM[2100952] USEC[2421780]
[0][2] init[1024,1024,1024] FP[3145728] INS[107087054] L2DCM[2124844] USEC[2421705]
[0][0] init[1024,1024,1024] FP[3145728] INS[123983711] L2DCM[2089784] USEC[2422204]
... tid 0 starting matrix multiply
... tid 1 starting matrix multiply
... tid 3 starting matrix multiply
... tid 2 starting matrix multiply

[1][1] init[1024,1024,1024] FP[3145728] INS[147571937] L2DCM[2220183] USEC[2541458]
[1][3] init[1024,1024,1024] FP[3145728] INS[109273232] L2DCM[2200654] USEC[2541429]
[1][2] init[1024,1024,1024] FP[3145728] INS[107456375] L2DCM[2214333] USEC[2541361]
[1][0] init[1024,1024,1024] FP[3145728] INS[144025387] L2DCM[2189125] USEC[2541868]
... tid 0 starting matrix multiply
... tid 1 starting matrix multiply
... tid 2 starting matrix multiply
... tid 3 starting matrix multiply
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
[0] mm[1024,1024,1024] FP[2151153664] INS[9020932602] L2DCM[270484334] USEC[11254098]
[0] mm[1024,1024,1024] FP[2151153664] INS[7525985513] L2DCM[270171124] USEC[11254286]
[0] mm[1024,1024,1024] FP[2151153664] INS[9273025751] L2DCM[270499158] USEC[11254282]
[0] [0] mm[1024,1024,1024] FP[2151153664] INS[7780969482] L2DCM[270106544] USEC[11254443]

[1] mm[1024,1024,1024] FP[2151153664] INS[9077245107] L2DCM[270414465] USEC[12324461]
[1] mm[1024,1024,1024] FP[2151153664] INS[7525968734] L2DCM[270386539] USEC[12324582]
[1] mm[1024,1024,1024] FP[2151153664] INS[9337961638] L2DCM[270389136] USEC[12324478]
[1] [0] mm[1024,1024,1024] FP[2151153664] INS[7628607535] L2DCM[270355014] USEC[12324730]
[0] [PAPI_TOT_INS] : Tot[ 68144406795 ] Hi[ 1 , 34078109945 ] Lo[ 0 , 34066296850 ] SDev[ 5906547.500000 ]
[0] [PAPI_FP_INS] : Tot[ 17234395136 ] Hi[ 0 , 8617197568 ] Lo[ 0 , 8617197568 ] SDev[ 0.000000 ]
[0] [PAPI_L2_DCM] : Tot[ 2180053564 ] Hi[ 1 , 1090369449 ] Lo[ 0 , 1089684115 ] SDev[ 342667.000000 ]
[0] [Real usec] : Tot[ 14866598 ] Hi[ 1 , 14866598 ] Lo[ 0 , 13676647 ] SDev[ 594975.500000 ]
THY: [1024,1024,1024] FP[ 17205035008 ]

Application 372242 resources: utime ~104s, stime ~12s
real    0m15.643s
user    0m0.344s
sys     0m0.052s

jaguarpf> export OMP_NUM_THREADS=16
jaguarpf> time aprun -n 2 -d 16 ./xzgemm-omp-krp 1024 1024 1024 64
[1][8] init[1024,1024] FP[786432] INS[79918946] L2DCM[659084] USEC[3046317]
[1][15] init[1024,1024,1024] FP[786432] INS[57756899] L2DCM[664614] USEC[3045422]
[1][9] init[1024,1024,1024] FP[786432] INS[59900799] L2DCM[662130] USEC[3046013]
[1][3] init[1024,1024,1024] FP[786432] INS[180090368] L2DCM[732026] USEC[3046349]
[1][1] init[1024,1024,1024] FP[786432] INS[194691159] L2DCM[723838] USEC[3047111]
[1][14] init[1024,1024,1024] FP[786432] INS[56535439] L2DCM[677196] USEC[3045744]
[1][6] init[1024,1024,1024] FP[786432] INS[175603943] L2DCM[749046] USEC[3046754]
[1][7] init[1024,1024,1024] FP[786432] INS[214880510] L2DCM[729561] USEC[3047047]
[1][10] init[1024,1024,1024] FP[786432] INS[43720487] L2DCM[675307] USEC[3045602]
[1][11] init[1024,1024,1024] FP[786432] INS[28589183] L2DCM[676626] USEC[3046085]
[1][13] init[1024,1024,1024] FP[786432] INS[55209462] L2DCM[666248] USEC[3045542]
[1][12] init[1024,1024,1024] FP[786432] INS[52903444] L2DCM[672625] USEC[3046347]
[1][2] init[1024,1024,1024] FP[786432] INS[191718131] L2DCM[739012] USEC[3047444]
[1][4] init[1024,1024,1024] FP[786432] INS[323208170] L2DCM[712739] USEC[3046725]
[1][5] init[1024,1024,1024] FP[786432] INS[392720235] L2DCM[700766] USEC[3047327]
[1][0] init[1024,1024,1024] FP[786432] INS[253845665] L2DCM[726203] USEC[3049019]
... tid 0 starting matrix multiply
... tid 1 starting matrix multiply
... tid 4 starting matrix multiply
... tid 11 starting matrix multiply
... tid 5 starting matrix multiply
... tid 6 starting matrix multiply
... tid 10 starting matrix multiply
... tid 7 starting matrix multiply
... tid 9 starting matrix multiply
... tid 8 starting matrix multiply
... tid 13 starting matrix multiply
... tid 12 starting matrix multiply
... tid 3 starting matrix multiply
... tid 2 starting matrix multiply
... tid 15 starting matrix multiply
... tid 14 starting matrix multiply

[0][2] init[1024,1024,1024] FP[786432] INS[31258021] L2DCM[803091] USEC[3171771]
[0][3] init[1024,1024,1024] FP[786432] INS[108059471] L2DCM[767536] USEC[3171712]
[0][1] init[1024,1024,1024] FP[786432] INS[67174896] L2DCM[778345] USEC[3172101]
[0][8] init[1024,1024,1024] FP[786432] INS[95220519] L2DCM[676076] USEC[3170732]
[0][9] init[1024,1024,1024] FP[786432] INS[107488417] L2DCM[667453] USEC[3171468]
[0][4] init[1024,1024,1024] FP[786432] INS[37627035] L2DCM[790858] USEC[3172147]
[0][5] init[1024,1024,1024] FP[786432] INS[82132310] L2DCM[771633] USEC[3171291]
[0][10] init[1024,1024,1024] FP[786432] INS[203736640] L2DCM[666570] USEC[3170467]
[0][11] init[1024,1024,1024] FP[786432] INS[132133434] L2DCM[673793] USEC[3170821]
[0][13] init[1024,1024,1024] FP[786432] INS[233945855] L2DCM[654129] USEC[3171565]
[0][12] init[1024,1024,1024] FP[786432] INS[137903081] L2DCM[677007] USEC[3171372]
[0][14] init[1024,1024,1024] FP[786432] INS[167255831] L2DCM[670586] USEC[3170859]
[0][15] init[1024,1024,1024] FP[786432] INS[142118241] L2DCM[681175] USEC[3171323]
[0][6] init[1024,1024,1024] FP[786432] INS[52790782] L2DCM[797942] USEC[3172227]
[0][7] init[1024,1024,1024] FP[786432] INS[73702671] L2DCM[784968] USEC[3172182]
[0][0] init[1024,1024,1024] FP[786432] INS[34818104] L2DCM[798948] USEC[3173355]
... tid 0 starting matrix multiply
... tid 1 starting matrix multiply
... tid 2 starting matrix multiply
... tid 5 starting matrix multiply
... tid 8 starting matrix multiply
... tid 9 starting matrix multiply
... tid 6 starting matrix multiply
... tid 3 starting matrix multiply
... tid 11 starting matrix multiply
... tid 10 starting matrix multiply
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
... tid 7 starting matrix multiply
... tid 12 starting matrix multiply
... tid 15 starting matrix multiply
... tid 4 starting matrix multiply
... tid 14 starting matrix multiply
... tid 13 starting matrix multiply

[1] mm[1024,1024,1024] FP[537788416] INS[1902328667] L2DCM[67542205] USEC[2450433]
[1] mm[1024,1024,1024] FP[537788416] INS[1902309615] L2DCM[67370255] USEC[2450407]
[1] mm[1024,1024,1024] FP[537788416] INS[1929383249] L2DCM[67616683] USEC[2451051]
[1] mm[1024,1024,1024] FP[537788416] INS[1883046687] L2DCM[67423282] USEC[2450759]
[1] mm[1024,1024,1024] FP[537788416] INS[1885429497] L2DCM[67445091] USEC[2451025]
[1] mm[1024,1024,1024] FP[537788416] INS[1957808606] L2DCM[67552387] USEC[2450517]
[1] mm[1024,1024,1024] FP[537788416] INS[1926013020] L2DCM[67479105] USEC[2451200]
[1] mm[1024,1024,1024] FP[537788416] INS[1929956596] L2DCM[67541220] USEC[2451097]
[1] mm[1024,1024,1024] FP[537788416] INS[1918318952] L2DCM[67533173] USEC[2450320]
[1] mm[1024,1024,1024] FP[537788416] INS[1937251820] L2DCM[67571747] USEC[2450273]
[1] mm[1024,1024,1024] FP[537788416] INS[1920716790] L2DCM[67563809] USEC[2450493]
[1] mm[1024,1024,1024] FP[537788416] INS[1953431820] L2DCM[67478355] USEC[2450742]
[1] mm[1024,1024,1024] FP[537788416] INS[1909530177] L2DCM[67529724] USEC[2450694]
[1] mm[1024,1024,1024] FP[537788416] INS[1924981763] L2DCM[67561086] USEC[2451364]
[1] mm[1024,1024,1024] FP[537788416] INS[1912517745] L2DCM[67528581] USEC[2451287]
[0] mm[1024,1024,1024] FP[537788416] INS[1901529102] L2DCM[67531167] USEC[2447567]
[0] mm[1024,1024,1024] FP[537788416] INS[1882743947] L2DCM[67524646] USEC[2447197]
[0] mm[1024,1024,1024] FP[537788416] INS[1942454985] L2DCM[67480460] USEC[2447467]
[0] mm[1024,1024,1024] FP[537788416] INS[1917112250] L2DCM[67487103] USEC[2446871]
[0] mm[1024,1024,1024] FP[537788416] INS[1923651198] L2DCM[67544819] USEC[2447644]
[0] mm[1024,1024,1024] FP[537788416] INS[1899341106] L2DCM[67525157] USEC[2447193]
[0] mm[1024,1024,1024] FP[537788416] INS[1913912266] L2DCM[67553512] USEC[2447519]
[0] mm[1024,1024,1024] FP[537788416] INS[1901496435] L2DCM[67518733] USEC[2447129]
[0] mm[1024,1024,1024] FP[537788416] INS[1947297099] L2DCM[67547971] USEC[2447697]
[0] mm[1024,1024,1024] FP[537788416] INS[1893670889] L2DCM[67557677] USEC[2447938]
[0] mm[1024,1024,1024] FP[537788416] INS[1937503608] L2DCM[67558986] USEC[2447497]
[0] mm[1024,1024,1024] FP[537788416] INS[1926362191] L2DCM[67587182] USEC[2447152]
[0] mm[1024,1024,1024] FP[537788416] INS[1926407014] L2DCM[67524472] USEC[2447666]
[0] mm[1024,1024,1024] FP[537788416] INS[1940513189] L2DCM[67544175] USEC[2447174]
[0] mm[1024,1024,1024] FP[537788416] INS[1935408886] L2DCM[67477146] USEC[2447591]
[1][0] mm[1024,1024,1024] FP[537788416] INS[1926419442] L2DCM[67499009] USEC[2451824]
[0][0] mm[1024,1024,1024] FP[537788416] INS[1894900219] L2DCM[67524495] USEC[2448394]

[ PAPI_TOT_INS ] : Tot[ 65470406987 ] Hi[ 1 , 33078737286 ] Lo[ 0 , 32391669701 ] SDev[ 343533792.50000 ]
[ PAPI_FP_INS ] : Tot[ 17234395136 ] Hi[ 0 , 8617197568 ] Lo[ 0 , 8617197568 ] SDev[ 0.000000 ]
[ PAPI_L2_DCM ] : Tot[ 2183550544 ] Hi[ 0 , 1092147811 ] Lo[ 1 , 1091402733 ] SDev[ 372539.000000 ]
[Real usec] : Tot[ 5621749 ] Hi[ 0 , 5621749 ] Lo[ 1 , 5500843 ] SDev[ 60453.000000 ]

THY: [1024,1024,1024] FP[ 17205035008 ]

Application 372243 resources: utime ~96s, stime ~88s
real    0m8.126s
user    0m0.368s
sys     0m0.040s
```

Compilation: Scenario 3 -GNU

```
jaguarpf> cat cmpl.zgemm-omp-krp-gnu
cc -c -DKRP -fopenmp zgemm-omp-krp.c ;
cc -c ${PAPI_INCLUDE_OPTS} krp-init-sum.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-init.c;
cc -c ${PAPI_INCLUDE_OPTS} krp-stat.c;
cc -o xzgemm-omp-krp zgemm-omp-krp.o krp-stat.o krp-init-sum.o krp-init.o ${PAPI_POST_LINK_OPTS}

jaguarpf> module swap PrgEnv-pgi PrgEnv-gnu
jaguarpf> module load papi
jaguarpf> source cmpl.zgemm-omp-krp-gnu
jaguarpf>
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

Run Examples: Scenario 3 -GNU

```
jaguarpf> time aprun -n 1 -d 4 ./xzgemm-omp-krp 512 512 512 128
... tid 1 starting matrix multiply
... tid 0 starting matrix multiply
... tid 3 starting matrix multiply
... tid 2 starting matrix multiply

... tid 3 after matrix multiply
[0][3] mm[512,512,512] FP[269352960] INS[2092417295] L2DCM[23080222] USEC[1284165]
... tid 2 after matrix multiply
[0][2] mm[512,512,512] FP[269352960] INS[2092417929] L2DCM[22994550] USEC[1284126]
... tid 1 after matrix multiply
[0][1] mm[512,512,512] FP[269352960] INS[2092419110] L2DCM[21567402] USEC[1284405]
[0][0] mm[512,512,512] FP[270532608] INS[2116246499] L2DCM[16867461] USEC[1839051]

[ PAPI_TOT_INS ] : Tot[ 8393500833 ] Hi[ 0 , 8393500833 ] Lo[ 0 , 8393500833 ] SDev[ 0.000000 ]
[ PAPI_FP_INS ] : Tot[ 1078591488 ] Hi[ 0 , 1078591488 ] Lo[ 0 , 1078591488 ] SDev[ 0.000000 ]
[ PAPI_L2_DCM ] : Tot[ 84509635 ] Hi[ 0 , 84509635 ] Lo[ 0 , 84509635 ] SDev[ 0.000000 ]
[Real usec] : Tot[ 1839051 ] Hi[ 0 , 1839051 ] Lo[ 0 , 1839051 ] SDev[ 0.000000 ]
THY: [512,512,512] FP[ 1076887552 ]

Application 376451 resources: utime ~6s, stime ~1s
real    0m2.435s
user    0m0.332s
sys     0m0.068s

jaguarpf> time aprun -n 2 -d 4 ./xzgemm-omp-krp 512 512 512 128
... tid 2 starting matrix multiply
... tid 1 starting matrix multiply
... tid 0 starting matrix multiply
... tid 3 starting matrix multiply
... tid 0 starting matrix multiply
... tid 1 starting matrix multiply
... tid 2 starting matrix multiply
... tid 3 starting matrix multiply

... tid 3 after matrix multiply
[1][3] mm[512,512,512] FP[269352960] INS[2092417693] L2DCM[22034585] USEC[1361306]
... tid 2 after matrix multiply
[1][2] mm[512,512,512] FP[269352960] INS[2092420116] L2DCM[21579778] USEC[1361605]
... tid 1 after matrix multiply
[1][1] mm[512,512,512] FP[269352960] INS[2092419568] L2DCM[21783591] USEC[1361521]
[1][0] mm[512,512,512] FP[270532608] INS[2116268806] L2DCM[17040655] USEC[1924012]
... tid 1 after matrix multiply
[0][1] mm[512,512,512] FP[269352960] INS[2092418948] L2DCM[27134899] USEC[1403721]
... tid 2 after matrix multiply
[0][2] mm[512,512,512] FP[269352960] INS[2092419581] L2DCM[25809014] USEC[1403669]
... tid 3 after matrix multiply
[0][3] mm[512,512,512] FP[269352960] INS[2092418965] L2DCM[25897739] USEC[1403614]
[0][0] mm[512,512,512] FP[270532608] INS[2116263193] L2DCM[22396928] USEC[1972385]

[ PAPI_TOT_INS ] : Tot[ 16787046870 ] Hi[ 1 , 8393526183 ] Lo[ 0 , 8393520687 ] SDev[ 2748.000000 ]
[ PAPI_FP_INS ] : Tot[ 2157182976 ] Hi[ 0 , 1078591488 ] Lo[ 0 , 1078591488 ] SDev[ 0.000000 ]
[ PAPI_L2_DCM ] : Tot[ 183677189 ] Hi[ 0 , 101238580 ] Lo[ 1 , 82438609 ] SDev[ 9399985.500000 ]
[Real usec] : Tot[ 1972385 ] Hi[ 0 , 1972385 ] Lo[ 1 , 1924012 ] SDev[ 24186.500000 ]
THY: [512,512,512] FP[ 2153775104 ]

Application 376455 resources: utime ~13s, stime ~3s
real    0m2.675s
user    0m0.340s
sys     0m0.060s
```

The following tables compare the measured results of various run instances against the theory for each run scenario.

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

Problem (3 PEs, 10 iterations)	INS	FP	L2DCM	Time[μ s]
P1: (m,l,n) (512,1024,512) P2: (m',l',n') (512 256 512)				
initialization	238512518441	11010048	340	12801680
work phase 1 THY:PEs*nits*(8.m.n.l + 12.m.n)	249967678724	64534609920 6.4518881E+10	3100363431	7327472
work phase 2 THY:PEs * nits * (8.mm.nn.ll + 12.mm.nn)	62726003055	16216227840 1.6200499E+10	10650240	1050097
Totals	551206200220	80761847808	3111014011	21179249
Theory (THY)		8.07193802E+10		

Table 15: Measured machine events of two sequential work phases (zgemms) in loop. The initialization time is skewed due to interactive input.

Problem (3 groups, 57PEs)	INS	FP	L2DCM	Time[μ s]
4096 2048 4096 4 4 40 10 Pes[16] grp[0]	i(3946417338) w(1533124302178)	i(134217856) w(2857327659840) thy(2750792335360)	i(2859) w(1229089599)	i(316716) w(31815913)
4096 4096 4096 4 4 40 3 Pes[16] grp[1]	i(5911477646) w(926228706335)	i(201326720) w(1714044273504) thy(1649871421440)	i(2918) w(732393724)	i(441768) w(18888393)
8192 8192 8192 5 5 40 1 pes[57] grp[2]	i(23642932421) w(2468935609381)	i(805306568) w(4570314965442) thy(4398851817472)	i(4088) w(1686575355)	i(1085416) w(31127848)
world: get subgroup (s) world: other (o)	s(114331433362) o(729215927863)	s(0) o(0)	s(5119) o(7988)	s(1849181) o(31817359)
Totals : group (grp) world(wrld) simple(smpl)	grp(4961789445299) wrld(843547361225) smpl(5739835192277)	grp(9142827749930) wrld(0) smpl(9142827749930)	grp(3648068543) wrld(13107) smpl(3660573793)	grp(32213264) wrld(33666540) smpl(33626130)

Table 16: Measured machine events of multiple parallel work phases (zgemms) in loop in distinct subgroups.

Problem PEs nt/PE m l n chnk	FP	INS	L2DCM	Time[μ s]
2 pes, 4nt/pe 1024,1024,1024,256	init()	init()	init()	init()
p0,t0	3145728	123983711	2089784	2422204
p0,t1	3145728	126814035	2107375	2421820
p0,t2	3145728	107087054	2124844	2421705
p0,t3	3145728	107498702	2100952	2421780
p1,t0	3145728	144025387	2189125	2541868
p1,t1	3145728	147571937	2220183	2541458
p1,t2	3145728	107456375	2214333	2541361
p1,t3	3145728	109273232	2200654	2541429
1024,1024,1024,256	work()	work()	work()	work()
p0,t0	2151153664	7780969482	270106544	11254443
p0,t1	2151153664	9020932602	270484334	11254098
p0,t2	2151153664	7525985513	270171124	11254286
p0,t3	2151153664	9273025751	270499158	11254282
p1,t0	2151153664	7628607535	270355014	12324730
p1,t1	2151153664	9077245107	270414465	12324461
p1,t2	2151153664	7525968734	270386539	12324582
p1,t3	2151153664	9337961638	270389136	12324478
Totals	17234395136 17205035008	68144406795	2180053564	14866598

Table 17: Measured machine events of threaded parallel work phase (zgemm).

0.6.3 Automatic instrumentation in GNU environment

The approach to automated instrumentation is most simply explained by example. First, we convert the working environment from the default PGI environment to the GNU variant, and then we load the Cray performance tools.

```
> module swap PrgEnv-pgi PrgEnv-gnu  
> module load perftools
```

Environment variables are defined for both the threaded features of the code, and the hardware events for the tool to target.

```
> export OMP_NUM_THREADS=4  
> export PAT_RT_HWPC=PAPI_TOT_CYC,PAPI_TOT_INS,PAPI_FP_INS
```

The compilation proceeds in two steps: a vanilla build, and then a second build that introduces the profiling instrumentation into the binary from the first build. Note that we instruct the tool to provide a performance profile for MPI, OpenMP, and I/O groups during execution for the desired events.

```
> cc -c -fopenmp -finstrument-functions zgemm-omp.c  
> cc -o xzgemm-omp-cai zgemm-omp.o -fopenmp -lm  
WARNING: CrayPat is saving object files from a temporary directory into  
directory '/ccs/home/roche/.craypat/xzgemm-omp-cai/22681'  
  
WARNING: the link is executed in a temporary directory which may not be  
accessible later for pat_build  
  
WARNING: the resulting executable was created in a temporary directory  
which may not be accessible later for pat_build  
  
> pat_build -g mpi -g omp -g io -w xzgemm-omp-cai xzgemm-omp-cai+pat
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
WARNING: Compiler-instrumented profiling is present in program 'xzgemm-omp-cai'  
- trace intercept routines will not be generated.
```

Job execution is as usual except that the instrumented binary is used. The instrumentation will collect data and save it for post analysis.

```
> time aprun -n 1 -d 4 ./xzgemm-omp-cai+pat 1024 1024 1024 256  
CrayPat/X: Version 5.3.0 Revision 8530 12/09/11 10:11:21  
pat[WARNING][0]: program is compiler-instrumented for GNU function tracing  
[0]... tid 1 starting matrix multiply  
[0]... tid 2 starting matrix multiply  
[0]... tid 3 starting matrix multiply  
[0]... tid 0 starting matrix multiply  
THY: [1024,1024,1024] FP[ 8602517504 ]  
Experiment data file written:  
xzgemm-omp-cai+pat+23276-574t.xf  
Application 375935 resources: utime ~39s, stime ~6s  
  
real    0m13.617s  
user    0m0.380s  
sys     0m0.040s
```

Now, generating the performance report requires a last step as illustrated here, and then one can look at the output. It is worth noting that the automated instrumentation in this example fails to accurately account for the contribution of the lightweight processes (threads). Indeed, we expect the floating point work for the problem instance to be 8602517504 but measure 2155872263. One notes the ratio is almost exactly 4 -the number of threads of execution in use for the execution instance. One should study the KRP versions to see the correct separation of the process and thread -based work. In Q3, we will explore our use of the tool to make certain we have not failed to exploit its potential capabilities.

```
> pat_report -o zgemm-omp-cai-1p4nt.txt xzgemm-omp-cai+pat+23276-574t.xf  
Processing step 5 of 5  
  
> cat zgemm-omp-cai-1p4nt.txt | more  
CrayPat/X: Version 5.3.0 Revision 8530 (xf 8240) 12/09/11 10:11:21
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

Number of PEs (MPI ranks): 1

Numbers of PEs per Node: 1

Numbers of Threads per PE: 4

Number of Cores per Socket: 16

Execution start time: Wed Mar 28 17:54:08 2012

System type and speed: x86_64 2200 MHz

Current path to data file:

/tmp/work/roche/kr-tsts/mm-study/c-api/xzgemm-omp-cai+pat+23276-574t.ap2 (RTS)

Notes for table 1:

Table option:

-O profile

Options implied by table option:

-d ti%@0.95,ti,imb_t,imb_t%,tr -b gr,fu,th=HIDE

Options for related tables:

-O profile_pe.th -O profile_th_pe

-O profile+src -O load_balance

-O callers -O callers+src

-O calltree -O calltree+src

The Total value for Time, Calls is the sum for the Group values.

The Group value for Time, Calls is the sum for the Function values.

The Function value for Time, Calls is the max for the Thread values.

(To specify different aggregations, see: pat_help report options s1)

This table shows only lines with Time% > 0.95.

(To set thresholds to zero, specify: -T)

Percentages at each level are of the Total for the program.

(For percentages relative to next level up, specify:

-s percent=r[elative])

Table 1: Profile by Function Group and Function

Time%	Time	Imb.	Imb.	Calls	Group
-------	------	------	------	-------	-------

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

		Time	Time%		Function	
					Thread=HIDE	
100.0%	12.213947	--	--	34.0	Total	

100.0%	12.213618	--	--	7.0	USER	

97.9%	11.963266	0.000000	0.0%	2.0	main	
2.0%	0.250346	0.000000	0.0%	1.0	exit	
=====						
0.0%	0.000134	0.000000	0.0%	3.0	PTHREAD	
0.0%	0.000120	0.000053	59.2%	2.0	STDIO	
0.0%	0.000038	--	--	4.0	OMP	
0.0%	0.000023	0.000000	0.0%	12.0	IO	
0.0%	0.000010	--	--	5.0	MPI	
0.0%	0.000004	0.000000	0.0%	1.0	MPI_SYNC	
=====						

Notes for table 2:

Table option:

-O profile+hwpc

Options implied by table option:

-d ti%@0.95,ti,imb_t,imb_t%,tr,P -b gr,fu,th=HIDE

Options for related tables:

-O profile_pe.th -O profile_th_pe

-O profile+src -O load_balance

-O callers -O callers+src

-O calltree -O calltree+src

The Total value for each data item is the sum for the Group values.

The Group value for each data item is the sum for the Function values.

The Function value for each data item is the max for the Thread values.

(To specify different aggregations, see: pat_help report options s1)

This table shows only lines with Time% > 0.95.

(To set thresholds to zero, specify: -T)

Percentages at each level are of the Total for the program.

(For percentages relative to next level up, specify:

-s percent=r[elative])

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

Table 2: Profile by Function Group and Function

Group / Function / Thread=HIDE

=====			
Total			

Time%			100.0%
Time		12.213947	secs
Imb. Time		--	secs
Imb. Time%		--	
Calls	3.506	/sec	34.0 calls
PAPI_TOT_INS	1037.779M/sec	10063040357	instr
PAPI_FP_INS	222.330M/sec	2155872263	ops
PAPI_TOT_CYC	9.697	secs	21332826724 cycles
Average Time per Call			0.359234 secs
CrayPat Overhead : Time	0.0%		
User time (approx)	12.214	secs	26870760748 cycles 100.0% Time
HW FP Ops / User time	222.330M/sec	2155872263	ops 1.3%peak(DP)
HW FP Ops / WCT	222.330M/sec		
HW FP Ops / Inst			21.4%
Instr per cycle			0.47 inst/cycle
MIPS	1037.78M/sec		
MFLOPS (aggregate)	222.33M/sec		
=====			
USER			

Time%			100.0%
Time		12.213618	secs
Imb. Time		--	secs
Imb. Time%		--	
Calls	0.722	/sec	7.0 calls
PAPI_TOT_INS	1037.784M/sec	10062998779	instr
PAPI_FP_INS	222.332M/sec	2155872263	ops
PAPI_TOT_CYC	9.697	secs	21332642521 cycles
Average Time per Call			1.744803 secs
CrayPat Overhead : Time	0.0%		
User time (approx)	12.214	secs	26870093782 cycles 100.0% Time
HW FP Ops / User time	222.332M/sec	2155872263	ops 1.3%peak(DP)
HW FP Ops / WCT	222.332M/sec		
HW FP Ops / Inst			21.4%
Instr per cycle			0.47 inst/cycle

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
MIPS           1037.78M/sec
MFLOPS (aggregate) 222.33M/sec
=====
USER / main
-----
Time%          97.9%
Time           11.963266 secs
Imb. Time      0.000000 secs
Imb. Time%     0.0%
Calls          0.206 /sec    2.0 calls
PAPI_TOT_INS   1037.779M/sec 10062844812 instr
PAPI_FP_INS    222.335M/sec  2155872263 ops
PAPI_TOT_CYC   9.697  secs   21332416848 cycles
Average Time per Call 5.981633 secs
CrayPat Overhead : Time 0.0%
User time (approx) 11.963  secs 26319311894 cycles 100.0% Time
HW FP Ops / User time 222.335M/sec  2155872263 ops  1.3%peak(DP)
HW FP Ops / WCT     222.335M/sec
HW FP Ops / Inst    21.4%
Instr per cycle    0.47 inst/cycle
MIPS           1037.78M/sec
MFLOPS (aggregate) 222.33M/sec
=====
USER / exit
-----
Time%          2.0%
Time           0.250346 secs
Imb. Time      0.000000 secs
Imb. Time%     0.0%
Calls          0.010M/sec   1.0 calls
PAPI_TOT_INS   1545.357M/sec 147820 instr
PAPI_FP_INS    0.000  secs   0 ops
PAPI_TOT_CYC   0.000  secs   210440 cycles
Average Time per Call 0.250346 secs
CrayPat Overhead : Time 0.0%
User time (approx) 0.250  secs 550769794 cycles 100.0% Time
HW FP Ops / User time 0 ops   0.0%peak(DP)
HW FP Ops / WCT
HW FP Ops / Inst    0.0%
Instr per cycle    0.70 inst/cycle
MIPS           1545.36M/sec
MFLOPS (aggregate) 0.00M/sec
=====
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

PTHREAD

Time%	0.0%	
Time	0.000134 secs	
Imb. Time	0.000000 secs	
Imb. Time%	0.0%	
Calls	0.140M/sec	3.0 calls
PAPI_TOT_INS	246.273M/sec	5289 instr
PAPI_FP_INS		0 ops
PAPI_TOT_CYC	0.000 secs	47245 cycles
Average Time per Call		0.000045 secs
CrayPat Overhead : Time	5.3%	
User time (approx)	0.000 secs	300174 cycles 100.0% Time
HW FP Ops / User time		0 ops 0.0%peak (DP)
HW FP Ops / WCT		
HW FP Ops / Inst		0.0%
Instr per cycle		0.11 inst/cycle
MIPS	246.27M/sec	
MFLOPS (aggregate)	0.00M/sec	

STDIO

Time%	0.0%	
Time	0.000120 secs	
Imb. Time	0.000053 secs	
Imb. Time%	59.2%	
Calls	0.087M/sec	2.0 calls
PAPI_TOT_INS	429.604M/sec	9843 instr
PAPI_FP_INS		0 ops
PAPI_TOT_CYC	0.000 secs	50405 cycles
Average Time per Call		0.000060 secs
CrayPat Overhead : Time	3.9%	
User time (approx)	0.000 secs	252451 cycles 100.0% Time
HW FP Ops / User time		0 ops 0.0%peak (DP)
HW FP Ops / WCT		
HW FP Ops / Inst		0.0%
Instr per cycle		0.20 inst/cycle
MIPS	429.60M/sec	
MFLOPS (aggregate)	0.00M/sec	

OMP

Time%	0.0%
-------	------

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

Time		0.000038 secs
Imb. Time		-- secs
Imb. Time%		--
Calls	0.232M/sec	4.0 calls
PAPI_TOT_INS	583.130M/sec	10070 instr
PAPI_FP_INS		0 ops
PAPI_TOT_CYC	0.000 secs	37990 cycles
Average Time per Call		0.000009 secs
CrayPat Overhead : Time	24.9%	
User time (approx)	0.000 secs	36111 cycles 100.0% Time
HW FP Ops / User time		0 ops 0.0%peak (DP)
HW FP Ops / WCT		
HW FP Ops / Inst		0.0%
Instr per cycle		0.27 inst/cycle
MIPS	583.13M/sec	
MFLOPS (aggregate)	0.00M/sec	
<hr/>		
IO		
<hr/>		
Time%		0.0%
Time		0.000023 secs
Imb. Time		0.000000 secs
Imb. Time%		0.0%
Calls	2.002M/sec	12.0 calls
PAPI_TOT_INS	240.571M/sec	1442 instr
PAPI_FP_INS		0 ops
PAPI_TOT_CYC	0.000 secs	13186 cycles
Average Time per Call		0.000002 secs
CrayPat Overhead : Time	120.3%	
User time (approx)	0.000 secs	48170 cycles 100.0% Time
HW FP Ops / User time		0 ops 0.0%peak (DP)
HW FP Ops / WCT		
HW FP Ops / Inst		0.0%
Instr per cycle		0.11 inst/cycle
MIPS	240.57M/sec	
MFLOPS (aggregate)	0.00M/sec	
<hr/>		
MPI		
<hr/>		
Time%		0.0%
Time		0.000010 secs
Imb. Time		-- secs
Imb. Time%		--

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

Calls	0.469M/sec	5.0 calls
PAPI_TOT_INS	1175.962M/sec	12531 instr
PAPI_FP_INS		0 ops
PAPI_TOT_CYC	0.000 secs	23442 cycles
Average Time per Call		0.000002 secs
CrayPat Overhead : Time	121.3%	
User time (approx)	0.000 secs	20878 cycles 100.0% Time
HW FP Ops / User time		0 ops 0.0%peak (DP)
HW FP Ops / WCT		
HW FP Ops / Inst		0.0%
Instr per cycle		0.53 inst/cycle
MIPS	1175.96M/sec	
MFLOPS (aggregate)	0.00M/sec	

MPI_SYNC

Time%	0.0%	
Time	0.000004 secs	
Imb. Time	0.000000 secs	
Imb. Time%	0.0%	
Calls	0.184M/sec	1.0 calls
PAPI_TOT_INS	443.153M/sec	2404 instr
PAPI_FP_INS		0 ops
PAPI_TOT_CYC	0.000 secs	11934 cycles
Average Time per Call		0.000004 secs
CrayPat Overhead : Time	53.7%	
User time (approx)	0.000 secs	9182 cycles 100.0% Time
HW FP Ops / User time		0 ops 0.0%peak (DP)
HW FP Ops / WCT		
HW FP Ops / Inst		0.0%
Instr per cycle		0.20 inst/cycle
MIPS	443.15M/sec	
MFLOPS (aggregate)	0.00M/sec	

Notes for table 3:

Table option:

-O read_stats

Options implied by table option:

-d rt,rb,rR,rd@,rC -b fi

The Total value for each data item is the sum for the File Name values.

(To specify different aggregations, see: pat_help report options s1)

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

This table shows only lines with Reads > 0.

Table 3: File Input Stats by Filename

Read Time	Read MBBytes	Read Rate MBytes/sec	Reads	Read B/Call	File Name
0.000049	0.000641	12.964756	12.0	56.00	Total
-----	-----	-----	-----	-----	-----
0.000049	0.000641	12.964756	12.0	56.00	_UnknownFile_
=====	=====	=====	=====	=====	=====

Notes for table 4:

Table option:

-O write_stats

Options implied by table option:

-d wt,wb,wR,wr@,wC -b fi

The Total value for each data item is the sum for the File Name values.

(To specify different aggregations, see: pat_help report options s1)

This table shows only lines with Writes > 0.

Table 4: File Output Stats by Filename

Write Time	Write MBBytes	Write Rate MBytes/sec	Writes	Write B/Call	File Name
0.000274	0.000183	0.669219	5.0	38.40	Total
-----	-----	-----	-----	-----	-----
0.000274	0.000183	0.669219	5.0	38.40	stdout
=====	=====	=====	=====	=====	=====

Notes for table 5:

Table option:

-O program_time

Options implied by table option:

-d pt,hm -b th

The Total value for Process HiMem (MBytes), Process Time is the max for the Thread values.

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

(To specify different aggregations, see: pat_help report options s1)

Table 5: Wall Clock Time, Memory High Water Mark

Process	Process	Thread
Time	HiMem	
	(MBytes)	
12.701278	117.316	Total

12.701278	117.316	thread.0
=====		

===== Additional details =====

Experiment: trace

Original path to data file:

/lustre/widow2/scratch/roche/kr-tsts/mm-study/c-api/xzgemm-omp-cai+pat+23276-574t.xf (RTS)

Original program:

/lustre/widow2/scratch/roche/kr-tsts/mm-study/c-api/xzgemm-omp-cai

Instrumented with:

pat_build -g mpi -g omp -g io -w xzgemm-omp-cai xzgemm-omp-cai+pat

Instrumented program: ./xzgemm-omp-cai+pat

Program invocation: ./xzgemm-omp-cai+pat 1024 1024 1024 256

Exit Status: 0 for 1 PE

CPU Family: 15h Model: 01h Stepping: 2

Core Performance Boost: Configured for 0 PEs
Capable for 1 PEs

Memory pagesize: 4096

Programming environment: GNU

Runtime environment variables:

MODULE_VERSION_STACK=3.2.6.6

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
GNU_VERSION=4.6.2
PBS_VERSION=TORQUE-2.5.9-snap.201111021154
PAPI_VERSION=4.2.0
XTOS_VERSION=4.0.30
PERFTOOLS_VERSION=5.3.0
CRAYPAT_ROOT_FOR_EVAL=/opt/cray/perf-tools/$PERFTOOLS_VERSION
MPICH_ABORT_ON_ERROR=1
MPICH_DIR=/opt/cray/mpt/5.4.1/xt/gemini/mpich2-gnu/46
ATP_POST_LINK_OPTS=-Wl,-L/opt/cray/atp/1.4.1/lib/ -Wl,-lAtpSigHandler -Wl,
--undefined=__atpHandlerInstall
GCC_VERSION=4.6.2
CRAY_MPICH2_VERSION=5.4.1
PAT_REPORT_PRUNE_NAME=__cray$mt_start__,__cray_hwpc_,f_crash_hwpc_,cstart,
__pat_,pat_region_,PAT_,OMP.slave_loop,slave_entry,_new_slave_entry,__libc_start_main,
_start,__start,start_thread
,__wrap_,UPCADIO_,_upc_,upc_,__caf_,__pgas_
PAPI_INTERFACE_VERSION=no-cuda
MODULE_VERSION=3.2.6.6
ASYNCPE_VERSION=5.05
PAT_RT_HWPC=PAPI_TOT_CYC
ATP_MRNET_COMM_PATH=/opt/cray/atp/1.4.1/bin/atp_mrnet_commnnode_wrapper
OMP_NUM_THREADS=4
LIBSCI_VERSION=11.0.04.4
ATP_HOME=/opt/cray/atp/1.4.1
```

Report time environment variables:

```
CRAYPAT_ROOT=/opt/cray/perf-tools/5.3.0
PAT_REPORT_PRUNE_NAME=__cray$mt_start__,__cray_hwpc_,f_crash_hwpc_,cstart,__pat_,pat_region_,PAT_,
,__wrap_,UPCADIO_,_upc_,upc_,__caf_,__pgas_
```

Report command line options: -o zgemm-omp-cai-1p4nt.txt

Operating system:

```
Linux 2.6.32.45-0.3.2_1.0400.6221-crav_gem_c #1 SMP Wed Sep 28 03:54:16 UTC 2011
```

Hardware performance counter events:

```
PAPI_TOT_INS Instructions completed
PAPI_FP_INS Floating point instructions
PAPI_TOT_CYC Total cycles
CYCLES_RTC User Cycles (approx, from rtc)
```

Estimated minimum overhead per call of a traced function,
which was subtracted from the data shown in this report

```
(for raw data, use the option: -s overhead=include):  
PAPI_TOT_INS 1582.095  
PAPI_FP_INS 0.000  
PAPI_TOT_CYC 1817.940  
CYCLES_RTC 5268.590  
Time 2.349 microsecs
```

Number of traced functions: 112

(To see the list, specify: -s traced_functions=show)

GPUs

Compiler Optimization Certainly more can be said about matrix-matrix multiplication. Perhaps it is important to note that this kernel rests at the heart of most dense linear algebra computations both in the sequential and parallel environments, and is under the hood of many sparse products to execute dense sub-blocks of work.

— to be continued in Q3 w/ programming model and compiler based optimization schemes

The use of CUPTI needs to be couple to the hybrid MPI + OpenMP programming model described. We will undertake this in Q3.

```
qsub -I -A csc091qmcpac -q debug -V -lfeature=gpu -lsize=16,walltime=59:00 -lgres=widow2%widow3
```

0.7 Application Inputs, Scripts, Environments

0.7.1 Drekar

Q2 Modules

```
pawlo@jaguarpf-ext3> module list
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

Currently Loaded Modulefiles:

```
1) modules/3.2.6.6
2) xe-sysroot/4.0.30.securitypatch.20110928
3) xtpe-network-gemini
4) pgi/12.1.0
5) xt-libsci/11.0.04.4
6) udreg/2.3.1-1.0400.3911.5.6.gem
7) ugni/2.3-1.0400.3912.4.29.gem
8) pmi/3.0.0-1.0000.8661.28.2807.gem
9) dmapp/3.2.1-1.0400.3965.10.12.gem
10) gni-headers/2.1-1.0400.3906.5.1.gem
11) xpmem/0.1-2.0400.29883.4.6.gem
12) xe-sysroot/4.0.30
13) xt-asyncpe/5.05
14) atp/1.4.1
15) PrgEnv-pgi/4.0.30
16) xt-mpich2/5.4.1
17) xtpe-interlagos
18) eswrap/1.0.9
19) lustredu/1.0
20) DefApps
21) altd/1.0
22) git/1.7.8
pawlo@jaguarpf-ext3>
```

Q2 Environment

```
pawlo@jaguarpf-ext2> env
LESSKEY=/etc/lesskey.bin
MODULE_VERSION_STACK=3.2.6.6
INFODIR=/usr/local/info:/usr/share/info:/usr/info
NNTPSERVER=news
GNU_VERSION=4.6.2
PAPI_POST_LINK_OPTS= -L/opt/cray/papi/4.2.0/perf_events/no-cuda/lib -lpapi
HOSTNAME=jaguarpf-ext2
CRAY_UDREG_INCLUDE_OPTS=-I/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/include
HDF5_DIR=/opt/cray/hdf5-parallel/1.8.7-gnu/46
XKEYSYMDB=/usr/share/X11/XKeysymDB
CRAY_SITE_LIST_DIR=/etc/opt/cray/modules
LIBRARYMODULES=/opt/modules/3.2.6.6/init/.librarymodules:acml:alps:apprentice2:
blcr:dmapp:fftw:ga:gni-headers:hdf5:libfast:mpich1:mrnet:netcdf:ntk:onesided:
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
petsc:petsc-complex:pmi:portals:rca:tpsl:trilinos:udreg:ugni:xpmem:xt-atp:  
xt-lgdb:xt-libsci:xt-mpt:xt-papi:/etc/opt/cray/modules/site_librarymodules  
PE_ENV=GNU  
PAPI_VERSION=4.2.0  
SHELL=/bin/bash  
TERM=xterm  
HOST=jaguarpf-ext2  
HISTSIZE=1000  
PROFILEREAD=true  
XTOS_VERSION=4.0.30  
HDF5_INCLUDE_OPTS=/opt/cray/hdf5-parallel/1.8.7/gnu/46/include  
SSH_CLIENT=134.253.112.64 42644 22  
CRAY_UGNI_POST_LINK_OPTS=-L/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/lib64  
CRAY_XPMEM_POST_LINK_OPTS=-L/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/lib64  
CRAY_MPICH2_DIR=/opt/cray/mpt/5.4.1/xt/gemini/mpich2-gnu/46  
ALTD_SELECT_OFF_USERS=  
ALTD_SELECT_ON=0  
MORE=-s1  
ALTD_VERBOSE=0  
SSH_TTY=/dev/pts/36  
USER=pawlo  
INTEL_PRE_COMPILE_OPTS= -msse3  
ASYNCPE_DIR=/opt/cray/xt-asyncpe/5.05  
BUILD_OPTS=/opt/cray/xt-asyncpe/5.05/bin/build-opts  
INTEL_NETCDF=120  
LD_LIBRARY_PATH=/opt/cray/papi/4.2.0/perf_events/no-cuda/lib:/opt/gcc/mpc/0.8.1/lib:  
/opt/gcc/mpfr/2.4.2/lib:/opt/gcc/gmp/4.3.2/lib:/opt/gcc/4.6.2/snobs/lib64:  
/opt/cray/atp/1.4.1/lib  
CRAY_HDF5_PARALLEL_VERSION=1.8.7  
XNLS_PATH=/usr/share/X11/nls  
WORKDIR=/tmp/work/pawlo  
ENV=/etc/bash.bashrc  
MPICH_DIR=/opt/cray/mpt/5.4.1/xt/gemini/mpich2-gnu/46  
ALTD_ON=1  
MPICH_ABORT_ON_ERROR=1  
HOSTTYPE=x86_64  
ATP_POST_LINK_OPTS=-Wl,-L/opt/cray/atp/1.4.1/lib/ -Wl,-lAtpSigHandler  
-Wl,--undefined=__atpHandlerInstall  
CRAY_PRGENVGNU=loaded  
RCLOCAL_PRGENV=true  
FROM_HEADER=  
CRAY_MPICH2_VERSION=5.4.1  
PE_PRODUCT_LIST=CRAY_HDF5:CRAY_NETCDF:ATP:ASYNCPE:XT_SYSROOT:CRAY_XPMEM:CRAY_DMAPP:
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
CRAY_PMI:CRAY_UGNI:CRAY_UDREG:LIBSCI:GNU:GCC:XTPE_INTERLAGOS:CRAY_MPICH2:PAPI
GCC_VERSION=4.6.2
PAPI_INTERFACE=perf_events
PAGER=less
FFTW_SYSTEM_WISDOM_DIR=/opt/xt-libsci/11.0.04.4
CSHEDIT=emacs
PAPI_INCLUDE_OPTS= -I/opt/cray/papi/4.2.0/perf_events/no-cuda/include
XDG_CONFIG_DIRS=/etc/xdg
MINICOM=-c on
USERMODULES=/opt/modules/3.2.6.6/init/.usermodules:acml:alps:apprentice2:atp:blcr:
cce:craypat:dmapp:fftw:ga:gcc:gni-headers:hdf5:intel:libfast:mpich1:mrnet:netcdf:
ntk:onesided:pathscale:petsc:petsc-complex:pgi:pmi:portals:PrgEnv-cray:PrgEnv-gnu:
PrgEnv-intel:PrgEnv-pathscale:PrgEnv-pgi:rca:stat:tpsl:trilinos:udreg:ugni:xpmem:
xt-asyncpe:xt-crayspat:xt-lgdb:xt-libsci:xt-mpt:xt-papi:xt-totalview:
/etc/opt/cray/modules/site_usermodules
CRAY_DMAPP_INCLUDE_OPTS=-I/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/include
-I/opt/cray/gni-headers/2.1-1.0400.3906.5.1.gem/include
PATH=/opt/cray/papi/4.2.0/perf_events/no-cuda/bin:
/opt/cray/hdf5-parallel/1.8.7-gnu/46/bin:/opt/cray/netcdf-hdf5parallel/4.1.3-gnu/46/bin:
/opt/cray/atp/1.4.1/bin:/opt/cray/xt-asyncpe/5.05/bin:
/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/bin:
/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/bin:
/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/bin:
/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/bin:/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/bin:
/opt/gcc/4.6.2/bin:/sw/xk6/cmake/2.8.6/cle4.0_gnu4.3.4/bin:
/sw/xk6/git/1.7.8/sles11.1_gnu4.3.4/bin:/sw/xk6/altd/bin:/sw/xk6/bin:
/sw/xk6/lustredu/1.0/sles11.1_gnu4.3.4/lustredu/bin:/opt/cray/eslogin/eswrap/1.0.9/bin:
/opt/modules/3.2.6.6/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/X11R6/bin:
/usr/games:/opt/bin:/usr/lib/mit/bin:/usr/lib/mit/sbin:/opt/dell/srvadmin/bin:/opt/bin:
/opt/public/bin:/opt/bin:/opt/public/bin
MAIL=/var/mail/pawlo
MODULE_VERSION=3.2.6.6
SYSROOT_GCC_VER=4.3.2
PAPI_INTERFACE_VERSION=no-cuda
CPU=x86_64
XTPE_NETWORK_TARGET=gemini
ESWRAP_LOGIN=jaguarpf-login5
CRAY_NETCDF=73
CRAY_HDF5=73
CRAY_HDF5_DIR=/opt/cray/hdf5-parallel/1.8.7
ASYNCPE_VERSION=5.05
CRAY_UDREG_POST_LINK_OPTS=-L/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/lib64
INTEL_HDF5=120
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
PWD=/tmp/proj/nfi007/pawlo/Trilinos_slow_fei
INPUTRC=/etc/inputrc
TARGETMODULES=/opt/modules/3.2.6.6/init/.targetmodules:craype-cpu-sandybridge:
craype-network-aries:xtpe-barcelona:xtpe-interlagos:xtpe-istanbul:xtpe-mc12:
xtpe-mc8:xtpe-netork-aries:xtpe-network-gemini:xtpe-network-seastar:
xtpe-shanghai:xtpe-target-cnl:xtpe-target-native:
/etc/opt/cray/modules/site_targetmodules
_LMFILES_= /opt/modulefiles/modules/3.2.6.6:
/opt/modulefiles/xe-sysroot/4.0.30.securitypatch.20110928:
/opt/cray/xt-asyncpe/default/modulefiles/xtpe-network-gemini:
/opt/cray/modulefiles/xt-mpich2/5.4.1:
/opt/cray/xt-asyncpe/default/modulefiles/xtpe-interlagos:
/opt/modulefiles/eswrap/1.0.9:/sw/xk6/modulefiles/lustredu/1.0:
/sw/xk6/modulefiles/DefApps:/sw/xk6/modulefiles/altd
/1.0:/sw/xk6/modulefiles/git/1.7.8:/sw/xk6/modulefiles/cmake/2.8.6:
/opt/modulefiles/gcc/4.6.2:/opt/cray/modulefiles/xt-libsci/11.0.04.4:
/opt/cray/gem/modulefiles/udreg/2.3.1-1.0400.3911.
5.6.gem:/opt/cray/gem/modulefiles/ugni/2.3-1.0400.3912.4.29.gem:
/opt/cray/gem/modulefiles/pmi/3.0.0-1.0000.8661.28.2807.gem:
/opt/cray/gem/modulefiles/dmapp/3.2.1-1.0400.3965.10.12.gem:
/opt/cray/gem/modulefiles/gni-headers/2.1-1.0400.3906.5.1.gem:
/opt/cray/gem/modulefiles/xpmem/0.1-2.0400.29883.4.6.gem:
/opt/modulefiles/xe-sysroot/4.0.30:/opt/modulefiles/xt-asyncpe/5.05:
/opt/cray/modulefiles/atp/1.4.1:/opt/modulefiles/PrgEnv-gnu/4.0.30:
/opt/cray/modulefiles/hdf5-parallel/1.8.7:
/opt/cray/modulefiles/netcdf-hdf5parallel/4.1.3:/opt/cray/modulefiles/papi/4.2
.0
LANG=en_US.UTF-8
SYSTEM_USERDIR=/tmp/work/pawlo
PGI_HDF5=109
PYTHONSTARTUP=/etc/pythonstart
MODULEPATH=/opt/cray/gem/modulefiles:/opt/cray/xt-asyncpe/default/modulefiles:
/opt/cray/modulefiles:/opt/modules/3.2.6.6/modulefiles:/opt/modulefiles:
/sw/xk6/modulefiles
LOADEDMODULES=modules/3.2.6.6:xe-sysroot/4.0.30.securitypatch.20110928:
xtpe-network-gemini:xt-mpich2/5.4.1:xtpe-interlagos:eswrap/1.0.9:lustredu/1.0:
DefApps:altd/1.0:git/1.7.8:cmake/2.8.6:gcc/4.6.2:xt-libsci/11.0.04.4:
udreg/2.3.1-1.0400.3911.5.6.gem:ugni/2.3-1.0400.3912.4.29.gem:
pmi/3.0.0-1.0000.8661.28.2807.gem:dmapp/3.2.1-1.0400.3965.10.12.gem:
gni-headers/2.1-1.0400.3906.5.1.gem:xpmem/0.1-2.0400.29883.4.6.gem:
xe-sysroot/4.0.30:xt-asyncpe/5.05:atp/1.4.1:PrgEnv-gnu/4.0.30:
hdf5-parallel/1.8.7:netcdf-hdf5parallel/4.1.3:papi/4.2.0
PATHSCALE_PRE_COMPILE_OPTS= -march=barcelona
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
SHMEM_ABORT_ON_ERROR=1
CRAY_DMAPP_POST_LINK_OPTS=-L/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/lib64
PERL5PATH=/sw/xk6/git/1.7.8/sles11.1_gnu4.3.4/lib/perl5/site_perl
CRAY_NETCDF_DIR=/opt/cray/netcdf-hdf5parallel/4.1.3
CRAY_PMI_POST_LINK_OPTS=-L/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/lib64
HOME=/ccs/home/pawlo
SHLVL=2
QT_SYSTEM_DIR=/usr/share/desktop-data
OSTYPE=linux
LESS_ADVANCED_PREPROCESSOR=no
NETCDF_DIR=/opt/cray/netcdf-hdf5parallel/4.1.3/gnu/46
ALTD_PATH=/sw/xk6/altd
LINKER_X86_64=/sw/xk6/altd/bin/ld
XCURSOR_THEME=DMZ
LS_OPTIONS=-N --color=none -T 0
CRAY_MPICH2_BASEDIR=/opt/cray/mpt/5.4.1/xt/gemini
CRAY_PMI_INCLUDE_OPTS=-I/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/include
WINDOWMANAGER=/usr/bin/gnome
PRGENVMODULES=/opt/modules/3.2.6.6/init/.prgenvmodules:PrgEnv-cray:PrgEnv-gnu:
PrgEnv-intel:PrgEnv-pathscale:PrgEnv-pgi
ATP_MRNET_COMM_PATH=/opt/cray/atp/1.4.1/bin/atp_mrnet_commnnode_wrapper
GCC_PATH=/opt/gcc/4.6.2
XTPE_INTERLAGOS_ENABLED=ON
LOGNAME=pawlo
MACHTYPE=x86_64-suse-linux
LESS=-M -I
G_FILENAME_ENCODING=@locale,UTF-8,ISO-8859-15,CP1252
ALTD_SELECT_USERS=
PYTHONPATH=/sw/xk6/git/1.7.8/sles11.1_gnu4.3.4/lib64/python2.6/site-package
CRAY_GNI_HEADERS_INCLUDE_OPTS=-I/opt/cray/gni-headers/2.1-1.0400.3906.5.1.gem/include
CVS_RSH=ssh
DMAPP_ABORT_ON_ERROR=1
SSH_CONNECTION=134.253.112.64 42644 160.91.205.199 22
XDG_DATA_DIRS=/usr/share:/etc/opt/kde3/share:/opt/kde3/share
TOOLMODULES=/opt/modules/3.2.6.6/init/.toolmodules:apprentice2:atp:craypat:mrnet:
stat:xt-crpat:xt-lgdb:xt-papi:xt-totalview:/etc/opt/cray/modules/site_toolmodules
PGI_NETCDF=109
MODULESHOME=/opt/modules/3.2.6.6
LESSOPEN=lessopen.sh %s
PKG_CONFIG_PATH=/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/lib64/pkgconfig:
/opt/cray/gni-headers/2.1-1.0400.3906.5.1.gem/lib64/pkgconfig:
/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/lib64/pkgconfig:
/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/lib64/pkgconfig:
```

```
/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/lib64/pkgconfig:  
/opt/cray/udreg/2.3.1-1.0400.3911.5.6.gem/lib64/pkgconfig  
LIBSCI_BASE_DIR=/opt/xt-libsci/11.0.04.4  
INFOPATH=/opt/gcc/4.6.2/snios/share/info:/usr/local/info:/usr/share/info:/usr/info  
CRAY_MPICH2_ROOTDIR=/opt/cray/mpt/5.4.1/xt  
LIBSCI_VERSION=11.0.04.4  
DISPLAY=localhost:19.0  
CRAY_PRE_COMPILE_OPTS=-hnetwork=gemini  
CRAY_CPU_TARGET=interlagos  
CRAY_NETCDF_HDF5PARALLEL_VERSION=4.1.3  
CRAY_UGNI_INCLUDE_OPTS=-I/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/include  
CRAY_XPMEM_INCLUDE_OPTS=-I/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/include  
SYSROOT_DIR=/opt/cray/xe-sysroot/4.0.30.securitypatch.20110928  
XAUTHLOCALHOSTNAME=jaguarpf-ext2  
LESSCLOSE=lessclose.sh %s %s  
GITDIR=/sw/xk6/git/1.7.8/sles11.1_gnu4.3.4  
ATP_HOME=/opt/cray/atp/1.4.1  
G_BROKEN_FILERAMES=1  
CRAY_LD_LIBRARY_PATH=/opt/cray/hdf5-parallel/1.8.7/gnu/46/lib:  
/opt/cray/netcdf-hdf5parallel/4.1.3/gnu/46/lib:  
/opt/cray/mpt/5.4.1/xt/gemini/mpich2-gnu/46/lib:  
/opt/cray/xpmem/0.1-2.0400.29883.4.6.gem/lib64:  
/opt/cray/dmapp/3.2.1-1.0400.3965.10.12.gem/lib64:  
/opt/cray/pmi/3.0.0-1.0000.8661.28.2807.gem/lib64:  
/opt/cray/ugni/2.3-1.0400.3912.4.29.gem/lib64:/opt/cray/udreg/2.3.1-  
1.0400.3911.5.6.gem/lib64:/opt/cray/mpt/5.4.1/xt/gemini/mpich2-pgi/109/lib  
COLORTERM=1  
_=~/usr/bin/env  
pawlo@jaguarpf-ext2:/tmp/proj/nfi007/pawlo/Trilinos_slow_fei>
```

Drekar is built using CMake. Specifically, it uses a build, test and integration system called Tribits which is a set of CMake extensions for handling complex package dependencies. This system was developed jointly by the Trilinos team at Sandia National Laboratories and the CASL VRI team at Oak Ridge National Laboratory. The build system requires 30 Trilinos packages which are automatically enabled and built as part of Drekar through the tribits build system. Additionally, 5 packages must be externally built by users: hdf5, netcdf, parmetis (3.2.1), boost (1.46.1), and papi. Versions of hdf5, netcdf, and papi are the default versions on jaguar and will be shown in the listing of the loaded modules.

Q2 Compilation

The required input for Drekar is (1) an xml input file and (2) an optional input mesh database

stored in the Genesis/Exodus II format. For the Joule metric runs, the mesh is generated inline so the input mesh database file is not required. The xml input file for run 10 is shown below.

A build directory containing the instructions and scripts for building Drekar on jaguarpf is contained within the Drekar source code in the directory:

```
Trilinos/Panzer/drekar/maintenance/build_jaguar/gnu/
```

A summary of the build steps follows. Files referenced in the build steps are in the directory above.

1. Load the prebuilt modules for jaguar. These are contained in the file “setup_drekar_env”.

```
pawlo@jaguarpf-ext3> cat setup_drekar_env
module unload PrgEnv-pgi
module unload PrgEnv-gnu
module load cmake
module load git
module load PrgEnv-gnu
module load netcdf-hdf5parallel
module load papi
pawlo@jaguarpf-ext3>
```

2. Download and untar the Boost library. Drekar uses the “headers-only” library so nothing need be compiled. Just untar the source and point to it in the Drekar configure script.
3. Build the parmetis library. During Q2 benchmarking, the parmetis library was not available on jaguarpf, so the drekar team built their own version. Version 3.2.0 of the library is required for interoperability with the Zoltan load balancing package and the ML algebraic multilevel preconditioner package. The Makefile.in file for Parmetis that exists in the top level directory for parmetis was configured as follows:

```
pawlo@jaguarpf-ext3> cat Makefile.in
```

```
# Which compiler to use
CC = cc
```

```
# What optimization level to use
#OPTFLAGS = -g
OPTFLAGS = -O3

# Include directories for the compiler
INCDIR =

# What options to be used by the compiler
#OPTIONS = -g
OPTIONS = -DNDEBUG

# Which loader to use
LD = CC

# In which directories to look for any additional libraries
LIBDIR =

# What additional libraries to link the programs with (eg., -lmpi)
#XTRALIBS = -lefence
#XTRALIBS = -ldmalloc

# What archiving to use
AR = ar rv

# What to use for indexing the archive
#RANLIB = ranlib
RANLIB = ar -ts

VERNUM = 3.2.0
pawlo@jaguarpf-ext3>
```

Note that during the Q2 benchmarking a version of parmetis was added to jaguarpf (4.0), but the version was incompatible the current releases of the Zoltan and ML packages that requires 3.1. An upgrade to support Parmetis 4.X has been requested.

A script recording of the parmetis build process can be found in the svn repository for this report in the file “files/parmetis_build_output.txt”.

4. Build the Trilinos and Drekar code. Drekar is treated as just another Trilinos package although it is really a terminal application. This simplifies the build process since building Trilinos will automatically build Drekar, alleviating the user from learning multiple build systems. The script used to invoke cmake for Trilinos/Drekar builds on jaguar follows.

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
pawlo@jaguarpf-ext3> cat build_drekar_jaguar.sh
#!/bin/bash
rm -rf CMakeCache.txt
cmake \
-D CMAKE_INSTALL_PREFIX="/home/rppawlo/trilinos_install" \
-D BUILD_SHARED_LIBS:BOOL=OFF \
-D TPL_FIND_SHARED_LIBS:BOOL=OFF \
-D Trilinos_LINK_SEARCH_START_STATIC:BOOL=ON \
-D Trilinos_ENABLE_TEUCHOS_TIME_MONITOR:BOOL=ON \
-D Trilinos_EXTRA_REPOSITORIES="Panzer" \
-D Trilinos_ENABLE_DEBUG=OFF \
-D Teuchos_ENABLE_LONG_LONG_INT:BOOL=ON \
-D Trilinos_ENABLE_ALL_PACKAGES:BOOL=OFF \
-D Trilinos_ENABLE_ALL_OPTIONAL_PACKAGES:BOOL=ON \
-D Trilinos_ENABLE_TESTS:BOOL=OFF \
-D Intrepid_ENABLE_DEBUG_INF_CHECK:BOOL=OFF \
-D Trilinos_ENABLE_Teko:BOOL=ON \
-D Trilinos_ENABLE_Drekar:BOOL=ON \
-D Drekar_ENABLE_TESTS:BOOL=ON \
-D TPL_ENABLE_Pthread:BOOL=OFF \
-D TPL_ENABLE_BinUtils:BOOL=OFF \
-D TPL_BLAS_LIBRARIES="/opt/xt-libsci/default/gnu/45/interlagos/lib/libsci_gnu.a" \
-D TPL_LAPACK_LIBRARIES="/opt/xt-libsci/default/gnu/45/interlagos/lib/libsci_gnu.a" \
-D TPL_ENABLE_MPI:BOOL=ON \
-D TPL_ENABLE_Boost:BOOL=ON \
-D Boost_INCLUDE_DIRS:FILEPATH="/tmp/proj/nfi007/pawlo/TPLs/boost_1_46_1" \
-D TPL_ENABLE_Netcdf:BOOL=ON \
-D Netcdf_INCLUDE_DIRS:PATH="/opt/cray/netcdf-hdf5parallel/4.1.2/gnu/45/include" \
-D Netcdf_LIBRARY_DIRS:PATH="/opt/cray/netcdf-hdf5parallel/4.1.2/gnu/45/lib" \
-D TPL_ENABLE_METIS:BOOL=ON \
-D METIS_INCLUDE_DIRS="/tmp/proj/nfi007/pawlo/TPLs/ParMetis-3.2.0/METISLib" \
-D METIS_LIBRARY_DIRS="/tmp/proj/nfi007/pawlo/TPLs/ParMetis-3.2.0" \
-D TPL_ENABLE_ParMETIS:BOOL=ON \
-D ParMETIS_INCLUDE_DIRS="/tmp/proj/nfi007/pawlo/TPLs/ParMetis-3.2.0" \
-D ParMETIS_LIBRARY_DIRS="/tmp/proj/nfi007/pawlo/TPLs/ParMetis-3.2.0" \
-D TPL_ENABLE_PAPI:BOOL=ON \
-D PAPI_INCLUDE_DIRS:FILEPATH="/opt/cray/papi/4.2.0/perf_events/no-cuda/include" \
-D PAPI_LIBRARY_DIRS:FILEPATH="/opt/cray/papi/4.2.0/perf_events/no-cuda/lib" \
-D CMAKE_CXX_COMPILER:FILEPATH="CC" \
-D CMAKE_C_COMPILER:FILEPATH="cc" \
-D CMAKE_Fortran_COMPILER:FILEPATH="ftn" \
-D CMAKE_CXX_FLAGS:STRING="-g -O3 -ansi -pedantic -ftrapv -Wall -Wno-long-long \
-Wno-strict-aliasing -DBOOST_NO_HASH -DMPICH_PTL_MATCH_OFF -DREDUCE_SCATTER_BUG" \
```

```
-D CMAKE_C_FLAGS:STRING="-g -O3 -DMPICH_PTL_MATCH_OFF -DREDUCE_SCATTER_BUG" \
-D CMAKE_Fortran_FLAGS:STRING="-g -O3 -DMPICH_PTL_MATCH_OFF -DREDUCE_SCATTER_BUG" \
-D CMAKE_VERBOSE_MAKEFILE:BOOL=OFF \
-D Trilinos_VERBOSE_CONFIGURE:BOOL=OFF \
-D CMAKE_SKIP_RULE_DEPENDENCY=ON \
-D Trilinos_ENABLE_STRONG_CXX_COMPILE_WARNINGS=OFF \
-D Trilinos_ENABLE_STRONG_C_COMPILE_WARNINGS=OFF \
-D Trilinos_ENABLE_SHADOW_WARNINGS=OFF \
-D CMAKE_BUILD_TYPE:STRING=Release \
./Trilinos
pawlo@jaguarpf-ext3>
```

Note that you must update this script to point to the installed parmetis and boost libraries.
Run this configure script in the BUILD directory.

Create an empty build directory outside the source code repository.

```
> mkdir BUILD
> cd BUILD
> ./build_drekar_jaguar.sh
```

A script recording output of the drekar configure process can be found in the svn repository
for this report in the file “files/drekar_configure_output.txt”.

5. Once configure has run successfully, Makefiles for the program will have been generated.
Then run a multithreaded make command:

```
> make drekar_mp -j12
```

This will produce the drekar_mp.exe executable. A script recording output of the drekar build
process can be found in the svn repository for this report in the file “files/drekar_build_output.txt”.

Q2 Input Settings

```
winterfell 19:13 > cat SwirlingJet_np131072_v10.xml
<ParameterList name="Drekar">

<ParameterList name="Mesh">
  <Parameter name="Source" type="string" value="Inline Mesh" />
  <ParameterList name="Inline Mesh">
    <Parameter name="Mesh Dimension" type="int" value="3" />
    <ParameterList name="Mesh Factory Parameter List">
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
<Parameter name="X Blocks" type="int" value="1" />
<Parameter name="Y Blocks" type="int" value="1" />
<Parameter name="Z Blocks" type="int" value="1" />
<Parameter name="X Procs" type="int" value="32" />
<Parameter name="Y Procs" type="int" value="32" />
<Parameter name="Z Procs" type="int" value="128" />
<Parameter name="X Elements" type="int" value="512" />
<Parameter name="Y Elements" type="int" value="512" />
<Parameter name="Z Elements" type="int" value="1024" />

<Parameter name="X0" type="double" value="-3.0" />
<Parameter name="Y0" type="double" value="-3.0" />
<Parameter name="Z0" type="double" value="0.0" />
<Parameter name="Xf" type="double" value="3.0" />
<Parameter name="Yf" type="double" value="3.0" />
<Parameter name="Zf" type="double" value="18.0" />
<ParameterList name="Periodic BCs">
<Parameter name="Count" type="int" value="2" />
    <Parameter name="Periodic Condition 1" type="string" value="yz-coord 1e-8: right;left" />
    <Parameter name="Periodic Condition 2" type="string" value="xz-coord 1e-8: top;bottom" />
</ParameterList>
</ParameterList>
</ParameterList>

<ParameterList name="Block ID to Physics ID Mapping">
    <Parameter name="eblock-0_0_0" type="string" value="fluid" />
</ParameterList>

<ParameterList name="Assembly">
    <Parameter name="Workset Size" type="unsigned long" value="1" />
</ParameterList>

<ParameterList name="Physics Blocks">

    <Parameter name="Physics Blocks" type="string" value="fluid" />

    <ParameterList name="fluid">

        <Parameter name="Number of Equation Sets" type="int" value="3" />

        <ParameterList name="EQ 0">
            <Parameter name="Name" type="string" value="Continuity" />
            <Parameter name="Basis" type="string" value="Q1" />
            <Parameter name="Integration Order" type="int" value="2" />
            <Parameter name="Model ID" type="string" value="global statistics model" />
            <Parameter name="Prefix" type="string" value="" />
            <ParameterList name="Options">
                <Parameter name="TAU_C" type="string" value="SHAKIB" />
                <Parameter name="PSPG STABILIZATION" type="string" value="ON" />
            </ParameterList>
        </ParameterList>

        <ParameterList name="EQ 1">
            <Parameter name="Name" type="string" value="Momentum" />
            <Parameter name="Basis" type="string" value="Q1" />
            <Parameter name="Integration Order" type="int" value="2" />
            <Parameter name="Model ID" type="string" value="fluid model" />
            <Parameter name="Prefix" type="string" value="" />
            <ParameterList name="Options">
                <Parameter name="TAU_M" type="string" value="SHAKIB" />
                <Parameter name="SUPG STABILIZATION" type="string" value="ON" />
            </ParameterList>
        </ParameterList>

        <ParameterList name="EQ 2">
            <Parameter name="Name" type="string" value="SARANS" />
            <Parameter name="Basis" type="string" value="Q1" />
            <Parameter name="Integration Order" type="int" value="2" />
            <Parameter name="Model ID" type="string" value="fluid model" />
            <Parameter name="Prefix" type="string" value="" />
            <ParameterList name="Options">
                <Parameter name="DCQ_NUHAT" type="string" value="OFF" />
                <Parameter name="TAU_NUHAT" type="string" value="SHAKIB" />
                <Parameter name="SUPG STABILIZATION" type="string" value="ON" />
                <Parameter name="USE SOURCE TERM" type="string" value="ON" />
            </ParameterList>
        </ParameterList>
    </ParameterList>
</ParameterList>
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
</ParameterList>

<ParameterList name="Closure Models">
    <ParameterList name="fluid model">
        <ParameterList name="DENSITY">
            <Parameter name="Value" type="double" value="1.0"/>
        </ParameterList>
        <ParameterList name="STRESS_TENSOR">
            <Parameter name="Value" type="string" value="NEWTONIAN"/>
        </ParameterList>
        <ParameterList name="VISCOSITY">
            <Parameter name="Value" type="string" value="SARANS"/>
            <Parameter name="Molecular Viscosity" type="double" value="1.0e-5"/>
        </ParameterList>
        <ParameterList name="SOURCE_MOMENTUM">
            <Parameter name="Value" type="string" value="BOUSSINESQ"/>
            <Parameter name="Reference Density" type="double" value="1.0"/>
            <Parameter name="Reference Temperature" type="double" value="1.0"/>
            <Parameter name="Volume Expansion Coefficient" type="double" value="1.0"/>
        </ParameterList>
        <ParameterList name="TEMPERATURE">
            <Parameter name="Value" type="double" value="1.0"/>
        </ParameterList>
        <ParameterList name="SOURCE_UX">
            <Parameter name="Value" type="double" value="0.0"/>
        </ParameterList>
        <ParameterList name="SOURCE_UY">
            <Parameter name="Value" type="double" value="0.0"/>
        </ParameterList>
        <ParameterList name="SOURCE_UZ">
            <Parameter name="Value" type="double" value="0.0"/>
        </ParameterList>
        <ParameterList name="GRAVITY_X">
            <Parameter name="Value" type="double" value="0.0"/>
        </ParameterList>
        <ParameterList name="GRAVITY_Y">
            <Parameter name="Value" type="double" value="0.0"/>
        </ParameterList>
        <ParameterList name="GRAVITY_Z">
            <Parameter name="Value" type="double" value="0.0"/>
        </ParameterList>
    </ParameterList>
</ParameterList>

<ParameterList name="magnetics model">
    <ParameterList name="PERMEABILITY">
        <Parameter name="Value" type="double" value="1.0"/>
    </ParameterList>
    <ParameterList name="RESISTIVITY">
        <Parameter name="Value" type="double" value="1.0"/>
    </ParameterList>
</ParameterList>

<ParameterList name="global statistics model">
    <ParameterList name="Global Statistics">
        <Parameter name="Value" type="string" value="UX,UY,UZ"/>
    </ParameterList>
</ParameterList>

</ParameterList>

<ParameterList name="User Data">
    <ParameterList name="Swirling3DJet">
        <Parameter name="R_jet" type="double" value="0.5"/>
        <Parameter name="omega" type="double" value="40.0"/>
        <Parameter name="V_z_jet" type="double" value="20.0"/>
        <Parameter name="V_z_block" type="double" value="10.0"/>
        <Parameter name="x_0" type="double" value="0.0"/>
        <Parameter name="y_0" type="double" value="0.0"/>
    </ParameterList>
</ParameterList>

</ParameterList name="Boundary Conditions">
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
<Parameter name="Number of Boundary Conditions" type="int" value="2"/>

<!-- Back -->
<ParameterList name="BC 0">
    <Parameter name="ID" type="unsigned long" value="0"/>
    <Parameter name="Type" type="string" value="Dirichlet"/>
    <Parameter name="Sideset ID" type="string" value="back"/>
    <Parameter name="Element Block ID" type="string" value="eblock-0_0_0"/>
    <Parameter name="Equation Set Name" type="string" value="" />
    <Parameter name="Strategy" type="string" value="Swirling3DJet"/>
    <ParameterList name="Data">
        <Parameter name="Closure Models" type="string" value="fluid model,magnetics model"/>
    </ParameterList>
</ParameterList>

<ParameterList name="BC 1">
    <Parameter name="ID" type="unsigned long" value="1"/>
    <Parameter name="Type" type="string" value="Dirichlet"/>
    <Parameter name="Sideset ID" type="string" value="back"/>
    <Parameter name="Element Block ID" type="string" value="eblock-0_0_0"/>
    <Parameter name="Equation Set Name" type="string" value="NUHAT"/>
    <Parameter name="Strategy" type="string" value="Constant"/>
    <ParameterList name="Data">
        <Parameter name="Value" type="double" value="1.0e-5"/>
    </ParameterList>
</ParameterList>

</ParameterList>

<ParameterList name="Initial Conditions">
    <ParameterList name="eblock-0_0_0">
        <ParameterList name="Initial Condition">
            <Parameter name="Value" type="string" value="SWIRLING3DJET"/>
        </ParameterList>
        <ParameterList name="NUHAT">
            <Parameter name="Value" type="double" value="1.0e-5"/>
        </ParameterList>
    </ParameterList>
</ParameterList>

<ParameterList name="Output">
    <Parameter name="File Name" type="string" value="output/swirling3DJet_131072_v10.exo"/>
</ParameterList>

<ParameterList name="Options">
    <Parameter name="Write Volume Assembly Graphs" type="bool" value="false"/>
    <Parameter name="Volume Assembly Graph Prefix" type="string" value="swirling3DJet_"/>
</ParameterList>

<ParameterList name="Solver Factories">
    <ParameterList name="Rythmos Observers">
        <Parameter name="Write Solution to Exodus File" type="string" value="ON"/>
        <Parameter name="Time Step Interval for Writing Solution" type="int" value="300"/>
        <Parameter name="Write Initial Condition" type="string" value="FALSE"/>
    </ParameterList>
</ParameterList>

<ParameterList name="Solution Control">
    <Parameter name="Piro Solver" type="string" value="Rythmos"/>
    <Parameter name="Compute Sensitivities" type="bool" value="0"/>
    <Parameter name="Jacobian Operator" type="string" value="Have Jacobian"/>
    <ParameterList name="LOCA">
        <ParameterList name="Bifurcation"/>
        <ParameterList name="Constraints"/>
        <ParameterList name="Predictor">
            <Parameter name="Method" type="string" value="Constant"/>
        </ParameterList>
        <ParameterList name="Stepper">
            <Parameter name="Continuation Method" type="string" value="Natural"/>
            <Parameter name="Initial Value" type="double" value="1.0"/>
            <Parameter name="Continuation Parameter" type="string" value="Parameter 0"/>
            <Parameter name="Max Steps" type="int" value="6"/>
            <Parameter name="Max Value" type="double" value="12.25"/>
            <Parameter name="Min Value" type="double" value="0.5"/>
            <Parameter name="Compute Eigenvalues" type="bool" value="1"/>
        </ParameterList>
        <ParameterList name="Eigensolver">
            <Parameter name="Method" type="string" value="Anasazi"/>
            <Parameter name="Operator" type="string" value="Shift-Invert"/>
            <Parameter name="Num Blocks" type="int" value="3"/>
            <Parameter name="Num Eigenvalues" type="int" value="1"/>
            <Parameter name="Block Size" type="int" value="1"/>
            <Parameter name="Maximum Restarts" type="int" value="0"/>
        </ParameterList>
    </ParameterList>
</ParameterList>
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
<Parameter name="Shift" type="double" value="1.0"/>
</ParameterList>
</ParameterList>
<!-- end "Stepper" -->
<ParameterList name="Step Size">
  <Parameter name="Initial Step Size" type="double" value="0.5"/>
  <Parameter name="Aggressiveness" type="double" value="2.0"/>
</ParameterList>
</ParameterList>
<!-- end "LOCA" -->
<ParameterList name="NOX">
  <ParameterList name="Direction">
    <Parameter name="Method" type="string" value="Newton"/>
    <ParameterList name="Newton">
      <Parameter name="Forcing Term Method" type="string" value="Constant"/>
      <Parameter name="Rescue Bad Newton Solve" type="bool" value="1"/>
      <ParameterList name="Linear Solver">
        <Parameter name="Tolerance" type="double" value="1e-3"/>
      </ParameterList>
    </ParameterList name="Stratimikos Linear Solver">
      <ParameterList name="NOX Stratimikos Options">
      </ParameterList>
    <ParameterList name="Stratimikos">
      <Parameter name="Linear Solver Type" type="string" value="AztecOO"/>
      <Parameter name="Preconditioner Type" type="string" value="Teko"/>
      <ParameterList name="Linear Solver Types">

        <ParameterList name="AztecOO">
          <ParameterList name="Forward Solve">
            <ParameterList name="AztecOO Settings">
              <Parameter name="Aztec Solver" type="string" value="GMRES"/>
              <Parameter name="Convergence Test" type="string" value="r0"/>
              <Parameter name="Size of Krylov Subspace" type="int" value="150"/>
              <Parameter name="Output Frequency" type="int" value="10"/>
            </ParameterList>
            <Parameter name="Max Iterations" type="int" value="150"/>
            <Parameter name="Tolerance" type="double" value="1e-5"/>
          </ParameterList>
        </ParameterList>
        <!-- end "AztecOO" -->
      <ParameterList name="Belos">
        <Parameter name="Solver Type" type="string" value="Block GMRES"/>
        <ParameterList name="Solver Types">
          <ParameterList name="Block GMRES">
            <Parameter name="Convergence Tolerance" type="double" value="1e-5"/>
            <Parameter name="Output Frequency" type="int" value="10"/>
            <Parameter name="Output Style" type="int" value="1"/>
            <Parameter name="Verbosity" type="int" value="33"/>
            <Parameter name="Maximum Iterations" type="int" value="200"/>
            <Parameter name="Block Size" type="int" value="1"/>
            <Parameter name="Num Blocks" type="int" value="200"/>
            <Parameter name="Flexible Gmres" type="bool" value="0"/>
          </ParameterList>
        </ParameterList>
        <ParameterList name="VerboseObject">
          <Parameter name="Output File" type="string" value="none"/>
          <Parameter name="Verbosity Level" type="string" value="medium"/>
        </ParameterList>
      </ParameterList>
      <!-- end "Belos" -->
    </ParameterList>
    <!-- end "Linear Solver Types" -->

    <ParameterList name="Preconditioner Types">

      <ParameterList name="Ifpack">
        <Parameter name="Overlap" type="int" value="1"/>
        <Parameter name="Prec Type" type="string" value="ILU"/>
        <ParameterList name="Ifpack Settings">
          <Parameter name="fact: drop tolerance" type="double" value="0"/>
          <Parameter name="fact: ilut level-of-fill" type="double" value="1"/>
          <Parameter name="fact: level-of-fill" type="int" value="1"/>
        </ParameterList>
      </ParameterList>
      <!-- end "Ifpack" -->

      <ParameterList name="ML">
        <Parameter name="Base Method Defaults" type="string" value="SA"/>
        <ParameterList name="ML Settings">
          <Parameter name="ML output" type="int" value="10"/>
          <Parameter name="prec type" type="string" value="MGW"/>
          <Parameter name="print unused" type="int" value="0"/>
        </ParameterList>
      </ParameterList>
    </ParameterList>
  </ParameterList>
</ParameterList>
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
<Parameter name="PDE equations" type="int" value="5"/>
<Parameter name="max levels" type="int" value="10"/>
<Parameter name="cycle applications" type="int" value="1"/>
<Parameter name="aggregation: threshold" type="double" value="0.0"/>
<Parameter name="aggregation: type" type="string" value="Uncoupled"/>
<Parameter name="aggregation: damping factor" type="double" value="0.0"/>
<Parameter name="aggregation: block scaling" type="bool" value="false"/>
<Parameter name="energy minimization: enable" type="bool" value="false"/>
<Parameter name="energy minimization: type" type="int" value="2"/>
<Parameter name="smoother: type" type="string" value="IFPACK"/>
<Parameter name="smoother: ifpack type" type="string" value="ILU"/>
<Parameter name="smoother: ifpack overlap" type="int" value="0"/>
<ParameterList name="smoother: ifpack list">
    <Parameter name="fact: level-of-fill" type="int" value="1"/>
    <Parameter name="schwarz: reordering type" type="string" value="rcm"/>
</ParameterList>
<Parameter name="smoother: damping factor" type="double" value="1.0"/>
<Parameter name="smoother: pre or post" type="string" value="both"/>
<Parameter name="smoother: sweeps" type="int" value="1"/>
<Parameter name="coarse: max size" type="int" value="500"/>
<Parameter name="coarse: type" type="string" value="Amesos-KLU"/>
</ParameterList>
</ParameterList>
<!-- End ML input -->

<ParameterList name="Teko">
<!-- comment out the stridding if using fully-coupled ML through Teko -->
<!-- <Parameter name="Strided Blocking" type="string" value="2 1"/> -->

<Parameter name="Inverse Type" type="string" value="ML-SARANS-fully-coupled"/>
<Parameter name="Write Block Operator" type="bool" value="false"/>
<Parameter name="Test Block Operator" type="bool" value="false"/>

<ParameterList name="Inverse Factory Library">

    <ParameterList name="ML-SARANS-fully-coupled">
        <ParameterList name="Required Parameters">
            <Parameter name="x-coordinates" type="string" value="none"/>
            <Parameter name="y-coordinates" type="string" value="none"/>
            <Parameter name="z-coordinates" type="string" value="none"/>
        </ParameterList>
        <Parameter name="Type" type="string" value="ML"/>
        <Parameter name="Base Method Defaults" type="string" value="SA"/>
        <ParameterList name="ML Settings">
            <Parameter name="ML output" type="int" value="10"/>
            <Parameter name="prec type" type="string" value="MGV"/>
            <Parameter name="print unused" type="int" value="0"/>
            <Parameter name="PDE equations" type="int" value="5"/>
            <Parameter name="max levels" type="int" value="10"/>
            <Parameter name="cycle applications" type="int" value="1"/>
            <Parameter name="aggregation: threshold" type="double" value="0.0"/>
            <Parameter name="aggregation: type" type="string" value="Uncoupled"/>
            <Parameter name="aggregation: damping factor" type="double" value="0.0"/>
            <Parameter name="aggregation: block scaling" type="bool" value="false"/>
            <Parameter name="energy minimization: enable" type="bool" value="false"/>
            <Parameter name="energy minimization: type" type="int" value="2"/>
            <Parameter name="smoother: type" type="string" value="IFPACK"/>
            <Parameter name="smoother: ifpack type" type="string" value="ILU"/>
            <Parameter name="smoother: ifpack overlap" type="int" value="1"/>
            <Parameter name="smoother: ifpack overlap (level 0)" type="int" value="0"/>
            <Parameter name="smoother: ifpack overlap (level 1)" type="int" value="0"/>
            <ParameterList name="smoother: ifpack list">
                <Parameter name="fact: level-of-fill" type="int" value="0"/>
                <Parameter name="schwarz: reordering type" type="string" value="rcm"/>
            </ParameterList>
            <Parameter name="smoother: damping factor" type="double" value="1.0"/>
            <Parameter name="smoother: pre or post" type="string" value="both"/>
            <Parameter name="smoother: sweeps" type="int" value="1"/>
            <Parameter name="coarse: max size" type="int" value="5000"/>
            <Parameter name="coarse: type" type="string" value="Amesos-KLU"/>
            <Parameter name="repartition: enable" type="int" value="1"/>
            <Parameter name="repartition: max min ratio" type="double" value="1.2"/>
            <Parameter name="repartition: min per proc" type="int" value="256"/>
            <Parameter name="repartition: partitioner" type="string" value="Zoltan"/>
            <Parameter name="repartition: Zoltan dimensions" type="int" value="3"/>
            <Parameter name="repartition: start level" type="int" value="2"/>
        </ParameterList>
    </ParameterList>
</ParameterList>
<!-- end "ML-SARANS-fully-coupled -->
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
<ParameterList name="LSC">
  <Parameter name="Type" type="string" value="NS LSC"/>
  <Parameter name="Inverse Type" type="string" value="Amesos"/>
  <Parameter name="Inverse Velocity Type" type="string" value="ML_GS-Velocity"/>
  <Parameter name="Inverse Pressure Type" type="string" value="ML_GS-Pressure"/>
  <Parameter name="Ignore Boundary Rows" type="bool" value="true"/>
  <Parameter name="Use LDU" type="bool" value="false"/>
  <Parameter name="Use Mass Scaling" type="bool" value="false"/> <!-- no masses available -->
  <Parameter name="Scaling Type" type="string" value="AbsRowSum"/>
</ParameterList>

<ParameterList name="SIMPLEX">
  <Parameter name="Type" type="string" value="NS SIMPLE"/>
  <!-- Inverse operations to use -->
  <Parameter name="Inverse Velocity Type" type="string" value="ML_GS-Velocity"/>
  <Parameter name="Inverse Pressure Type" type="string" value="ML_GS-Pressure"/>
  <Parameter name="Explicit Velocity Inverse Type" type="string" value="AbsRowSum"/>
  <Parameter name="Alpha" type="double" value="0.9"/>
</ParameterList>
<ParameterList name="ML_GS-Pressure">
  <Parameter name="Type" type="string" value="ML"/>
  <Parameter name="Base Method Defaults" type="string" value="SA"/>
  <ParameterList name="ML Settings">
    <Parameter name="ML output" type="int" value="0"/>
    <Parameter name="ML label" type="string" value="Pressure"/>
    <Parameter name="prec type" type="string" value="MGV"/>
    <Parameter name="print unused" type="int" value="0"/>
    <Parameter name="PDE equations" type="int" value="1"/>
    <Parameter name="max levels" type="int" value="10"/>
    <Parameter name="cycle applications" type="int" value="1"/>
    <Parameter name="aggregation: threshold" type="double" value="0.0"/>
    <Parameter name="aggregation: type" type="string" value="Uncoupled"/>
    <Parameter name="smoother: type" type="string" value="Gauss-Seidel"/>
    <Parameter name="smoother: pre or post" type="string" value="both"/>
    <Parameter name="smoother: sweeps" type="int" value="2"/>
    <Parameter name="coarse: max size" type="int" value="15000"/>
    <Parameter name="coarse: type" type="string" value="Amesos-KLU"/>
  </ParameterList>
</ParameterList>
<!-- end "ML_GS-Pressure" -->

<ParameterList name="ML_GS-Velocity">
  <ParameterList name="Required Parameters">
    <Parameter name="x-coordinates" type="string" value="none"/>
    <Parameter name="y-coordinates" type="string" value="none"/>
  </ParameterList>
  <Parameter name="Type" type="string" value="ML"/>
  <Parameter name="Base Method Defaults" type="string" value="SA"/>
  <ParameterList name="ML Settings">
    <Parameter name="ML output" type="int" value="10"/>
    <Parameter name="ML label" type="string" value="Velocity"/>
    <Parameter name="prec type" type="string" value="MGV"/>
    <Parameter name="print unused" type="int" value="0"/>
    <Parameter name="PDE equations" type="int" value="2"/>
    <Parameter name="max levels" type="int" value="10"/>
    <Parameter name="cycle applications" type="int" value="1"/>
    <Parameter name="aggregation: threshold" type="double" value="0.0"/>
    <Parameter name="aggregation: type" type="string" value="Uncoupled"/>
    <!--<Parameter name="aggregation: block scaling" type="bool" value="true"/>
    <Parameter name="energy minimization: enable" type="bool" value="true"/>
    <Parameter name="energy minimization: type" type="int" value="2"/>-->
    <Parameter name="smoother: type" type="string" value="block Gauss-Seidel"/>
    <Parameter name="smoother: pre or post" type="string" value="both"/>
    <Parameter name="smoother: sweeps" type="int" value="2"/>
    <Parameter name="coarse: max size" type="int" value="200"/>
    <Parameter name="coarse: type" type="string" value="Amesos-KLU"/>
    <Parameter name="repartition: enable" type="int" value="1"/>
    <Parameter name="repartition: max min ratio" type="double" value="1.2"/>
    <Parameter name="repartition: min per proc" type="int" value="256"/>
    <Parameter name="repartition: partitioner" type="string" value="Zoltan"/>
    <Parameter name="repartition: Zoltan dimensions" type="int" value="2"/>
  </ParameterList>
</ParameterList>
<!-- end "ML_GS-Velocity" -->

</ParameterList>
<!-- End "Inverse Factory Library" -->

</ParameterList>
<!-- end "Teko" -->
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
</ParameterList>
<!-- end "Preconditioner Types" -->

</ParameterList>
<!-- end "Stratimikos" -->

</ParameterList>
<!-- end "Stratimikos Linear Solver" -->

</ParameterList>
<!-- end "Newton" -->

</ParameterList>
<!-- end "Direction" -->
<ParameterList name="Line Search">
  <ParameterList name="Full Step">
    <Parameter name="Full Step" type="double" value="1"/>
  </ParameterList>
  <Parameter name="Method" type="string" value="Full Step"/>
</ParameterList>
<!-- end "Line Search" -->
<Parameter name="Nonlinear Solver" type="string" value="Line Search Based"/>
<ParameterList name="Printing">
  <Parameter name="Output Precision" type="int" value="3"/>
  <Parameter name="Output Processor" type="int" value="0"/>
  <ParameterList name="Output Information">
    <Parameter name="Error" type="bool" value="1"/>
    <Parameter name="Warning" type="bool" value="1"/>
    <Parameter name="Inner Iteration" type="bool" value="1"/>
    <Parameter name="Outer Iteration" type="bool" value="1"/>
    <Parameter name="Outer Iteration StatusTest" type="bool" value="1"/>
    <Parameter name="Test Details" type="bool" value="1"/>
    <Parameter name="Parameters" type="bool" value="1"/>
    <Parameter name="Details" type="bool" value="1"/>
    <Parameter name="Linear Solver Details" type="bool" value="1"/>
    <Parameter name="Stepper Iteration" type="bool" value="1"/>
    <Parameter name="Stepper Details" type="bool" value="1"/>
    <Parameter name="Stepper Parameters" type="bool" value="1"/>
  </ParameterList>
</ParameterList>
<!-- end "Printing" -->

<ParameterList name="Solver Options">
  <Parameter name="Status Test Check Type" type="string" value="Complete"/>
</ParameterList>
<ParameterList name="Status Tests">
  <Parameter name="Test Type" type="string" value="Combo"/>
  <Parameter name="Combo Type" type="string" value="OR"/>
  <Parameter name="Number of Tests" type="int" value="3"/>
  <ParameterList name="Test 0">
    <Parameter name="Test Type" type="string" value="Combo"/>
    <Parameter name="Combo Type" type="string" value="AND"/>
    <Parameter name="Number of Tests" type="int" value="2"/>
    <ParameterList name="Test 0">
      <Parameter name="Test Type" type="string" value="NormF"/>
      <Parameter name="Tolerance" type="double" value="1.0e-3"/>
    </ParameterList>
    <ParameterList name="Test 1">
      <Parameter name="Test Type" type="string" value="NormWRMS"/>
      <Parameter name="Tolerance" type="double" value="1.0"/>
      <Parameter name="Relative Tolerance" type="double" value="1.0e-3"/>
      <Parameter name="Absolute Tolerance" type="double" value="1.0e-5"/>
      <Parameter name="BDF Multiplier" type="double" value="1.0"/>
      <Parameter name="alpha" type="double" value="1.0"/>
      <Parameter name="beta" type="double" value="0.5"/>
    </ParameterList>
  </ParameterList>
  <ParameterList name="Test 1">
    <Parameter name="Test Type" type="string" value="MaxIter"/>
    <Parameter name="Maximum Iterations" type="int" value="10"/>
  </ParameterList>
  <ParameterList name="Test 2">
    <Parameter name="Test Type" type="string" value="FiniteValue"/>
  </ParameterList>
</ParameterList>
<!-- end "Status Tests" -->
</ParameterList>
<!-- end "NOX" -->

<ParameterList name="Rythmos">
  <Parameter name="Nonlinear Solver Type" type="string" value="NOX"/>
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
<Parameter name="Final Time" type="double" value="0.1"/> <!-- Was set to 1.0 -->
<Parameter name="Stepper Type" type="string" value="BDF"/>
<Parameter name="Step Control Strategy Type" type="string" value="ImplicitBDFRamping"/>
<Parameter name="Rythmos Integration Control Strategy" type="string" value="Simple"/>
<ParameterList name="Rythmos Stepper">
  <ParameterList name="VerboseObject">
    <Parameter name="Verbosity Level" type="string" value="medium"/>
  </ParameterList>
</ParameterList>
<ParameterList name="Stratimikos">
</ParameterList>
<ParameterList name="Rythmos Integrator">
  <ParameterList name="VerboseObject">
    <Parameter name="Verbosity Level" type="string" value="high"/>
  </ParameterList>
</ParameterList>
<ParameterList name="Rythmos Integration Control">
</ParameterList>
<ParameterList name="Rythmos Step Control Strategy">
  <Parameter name="Number of Constant First Order Steps" type="int" value="10"/>
  <Parameter name="Initial Step Size" type="double" value="1.0e-3"/>
  <Parameter name="Min Step Size" type="double" value="1.0e-7"/>
  <Parameter name="Max Step Size" type="double" value="1.0e+0"/>
  <Parameter name="Step Size Decrease Factor" type="double" value="0.5"/>
  <Parameter name="Step Size Increase Factor" type="double" value="1.2"/>
  <Parameter name="Max Order" type="int" value="1"/>
  <Parameter name="Absolute Error Tolerance" type="double" value="1.0e-5"/>
  <Parameter name="Relative Error Tolerance" type="double" value="1.0e-3"/>
</ParameterList>
</ParameterList>
<!-- end "Rythmos" -->
</ParameterList>
<!-- end "Solution Control" -->

</ParameterList>
<!-- end "Drekar" -->
```

Q2 Run Script

```
#!/usr/bin/csh
#PBS -A CSC091drekar
#PBS -N Drekar_SJ_131072_v10
#PBS -j oe
#PBS -l walltime=12:00:00,size=131072

setenv MPICH_PTL_MEMD_LIMIT 12000
setenv MPICH_PTL_OTHER_EVENTS 12000

cd $PBS_O_WORKDIR
date
time aprun -n 131072 ./drekar_mp.exe --i=SwirlingJet_np131072_v10.xml
```

0.7.2 Materials Project - VASP

Q2 Modules *jaguarpf.ccs.ornl.gov*

Currently Loaded Modulefiles:

- 1) modules/3.2.6.6
- 2) lustredu/1.0
- 3) DefApps
- 4) altd/1.0
- 5) torque/2.4.1b1-snap.200905191614
- 6) moab/6.1.3
- 7) nodestat/2.2-1.0400.29866.4.3.gem
- 8) sdb/1.0-1.0400.30000.6.18.gem
- 9) MySQL/5.0.64-1.0000.4667.20.1
- 10) lustre-cray_gem_s/1.8.4_2.6.32.45_0.3.2_1.0400.6221.1.1-1.0400.30303.0.0novmap1
- 11) udreg/2.3.1-1.0400.3911.5.6.gem
- 12) ugni/2.3-1.0400.3912.4.29.gem
- 13) gni-headers/2.1-1.0400.3906.5.1.gem
- 14) dmapp/3.2.1-1.0400.3965.10.12.gem
- 15) xpmem/0.1-2.0400.29883.4.6.gem
- 16) hss-l1m/6.0.0
- 17) Base-opts/1.0.2-1.0400.29823.8.1.gem
- 18) xtpe-network-gemini
- 19) pgi/12.1.0
- 20) xt-libsci/11.0.04.4
- 21) pmi/3.0.0-1.0000.8661.28.2807.gem
- 22) xt-asyncpe/5.05
- 23) atp/1.4.1
- 24) PrgEnv-pgi/4.0.30
- 25) xt-mpich2/5.4.1
- 26) xtpe-interlagos
- 27) fftw/3.3.0.0
- 28) subversion/1.6.17
- 29) papi/4.2.0

Q2 Modules *hopper.nersc.gov*

Currently Loaded Modulefiles:

- 1) modules/3.2.6.6
- 2) xtpe-network-gemini
- 3) pgi/11.9.0
- 4) xt-libsci/11.0.05
- 5) udreg/2.3.1-1.0400.3911.5.13.gem
- 6) ugni/2.3-1.0400.4127.5.20.gem
- 7) pmi/3.0.0-1.0000.8661.28.2807.gem
- 8) dmapp/3.2.1-1.0400.3965.10.63.gem
- 9) gni-headers/2.1-1.0400.4156.6.1.gem
- 10) xpmem/0.1-2.0400.30792.5.6.gem
- 11) xe-sysroot/4.0.36
- 12) xt-asyncpe/5.07
- 13) atp/1.4.2
- 14) PrgEnv-pgi/4.0.36
- 15) eswrap/1.0.10
- 16) xtpe-mc12
- 17) xt-shmem/5.4.4
- 18) xt-mpich2/5.4.4
- 19) torque/2.5.9
- 20) moab/6.1.4
- 21) fftw/3.2.2.1
- 22) subversion/1.6.4
- 23) papi/4.2.0

Q2 Compilation

We used the standard production programming environment and PGI compilers on both hopper and jaguar . Below we give the build script, vasp library and main program makefiles. The source code used corresponds to VASP 5.2.12, the latest available from Vienna in Q2.

```
build\_nccs.sh:  
#!/bin/tcsh  
module load fftw/3.3.0.0  
module load subversion  
module load papi  
module list  
cd vasp.5.lib  
make -f makefile.xt clean  
make -f makefile.xt  
cd ../vasp.5.2.12
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
foreach suffix (xt_vanilla_papi)
make -f makefile.${suffix} clean
make -f makefile.${suffix} vasp.${suffix}

if ( ! -e vasp.${suffix} ) then
    echo === FAILED to build vasp.${suffix}
    exit 1
endif

end
```

makefile xt:

```
.SUFFIXES: .inc .f .F
#-----
# Makefile for LINUX NAG f90
#-----
# fortran compiler
FC=ftn -target=linux

# C-preprocessor
CPP      = cpp -E -P -DLONGCHAR -C $*.F >$*.f

CFLAGS = -O -Mlist -target=linux
FFLAGS = -O1
FREE   = -Mfree

DOBJ = preclib.o timing_.o derrf_.o dclock_.o diolib.o dlexlib.o drdatab.o

#-----
# general rules
#-----

#libdmy.a: $(DOBJ) lapack_double.o
#Minor (avoidable) use made of linpack by elpol.F
libdmy.a: $(DOBJ) linpack_double.o lapack_double.o
-rm -f libdmy.a
ar vq libdmy.a $(DOBJ)
```

```
# files which do not require autodouble
lapack_double.o: lapack_double.f
$(FC) $(FFLAGS) $(NOFREE) -c lapack_double.f
lapack_single.o: lapack_single.f
$(FC) $(FFLAGS) $(NOFREE) -c lapack_single.f
lapack_cray.o: lapack_cray.f
$(FC) $(FFLAGS) $(NOFREE) -c lapack_cray.f
linpack_double.o: linpack_double.f
$(FC) $(FFLAGS) $(NOFREE) -c linpack_double.f
linpack_single.o: linpack_single.f
$(FC) $(FFLAGS) $(NOFREE) -c linpack_single.f

.c.o:
$(CC) $(CFLAGS) -c $*.c
.F.o:
$(CPP)
$(FC) $(FFLAGS) $(FREE) $(INCS) -c $*.f
.F.f:
$(CPP)
.f.o:
$(FC) $(FFLAGS) $(FREE) $(INCS) -c $*.f

clean:
rm -f *~ *.o *.lst *.a
```

makefile_xt_vanilla_papi:

```
.SUFFIXES: .inc .f .f90 .F
#
#
# This is a vanilla makefile for vanilla source for vasp 5.2.x
#
# Paul Kent / Oak Ridge National Laboratory
#
#-----
# Originally based on the Linux/PGI Vienna makefiles
#-----

# all CPP processed fortran files have the extension .f90
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
SUFFIX=.f90

#-----
# fortran compiler and linker
#-----
#FC=pgf90
FC=ftn
# fortran linker
FCL=$(FC)

SVNDEF = -D' SVN_REV="$$(shell svnversion -n .)"'

#-----
# whereis CPP ?? (I need CPP, can't use gcc with proper options)
# that's the location of gcc for SUSE 5.3
#
# CPP_    = /usr/lib/gcc-lib/i486-linux/2.7.2/cpp -P -C
#
# that's probably the right line for some Red Hat distribution:
#
# CPP_    = /usr/lib/gcc-lib/i386-redhat-linux/2.7.2.3/cpp -P -C
#
# SUSE X.X, maybe some Red Hat distributions:

CPP_ = ./preprocess <$*.F | /usr/bin/cpp -P -C -traditional >$*${SUFFIX}

#-----
# possible options for CPP:
# possible options for CPP:
# NGXhalf      charge density reduced in X direction
# wNGXhalf     gamma point only reduced in X direction
# avoidalloc   avoid ALLOCATE if possible
# PGF90        work around some for some PGF90 / IFC bugs
# CACHE_SIZE   1000 for PII,PIII, 5000 for Athlon, 8000-12000 P4, PD
# RPROMU_DGEMV use DGEMV instead of DGEMM in RPRO (depends on used BLAS)
# RACCMU_DGEMV use DGEMV instead of DGEMM in RACC (depends on used BLAS)
#-----

# See later for MPI version
#CPP      = $(CPP_) -DHOST=\"Cray XT/LinuxPgi\" \
#           -Dkind8 -DNGXhalf -DCACHE_SIZE=2000 -DPGF90 -Davoidalloc \
#           -DRPROMU_DGEMV
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
#-----
# general fortran flags (there must a trailing blank on this line)
# byterecl is strictly required for ifc, since otherwise
# the WAVECAR file becomes huge
#-----

#FFLAGS = -Mfree -Mx,119,0x200000
FFLAGS = -Mfree

#-----
# optimization,
# we have tested whether higher optimisation improves
# the performance, and found no improvements with -O3-5 or -fast
# (even on Athlon system, Athlon specific optimistation worsens performance)
#-----

#OFLAG = -O2 -tp p6
#OFLAG = -O2
#OFLAG = -g -O0

OFLAG_HIGH = $(OFLAG)
OBJ_HIGH =
#PK 20101215 wave_high very risky under optimization with many compilers (ifort, pgf90..)
OBJ_NOOPT = wave_high.o
DEBUG = -g -O0
INLINE = $(OFLAG)

#-----
# the following lines specify the position of BLAS and LAPACK
# VASP works fastest with the libgoto library
# so that's what we recommend
# PK: We use Cray Libsci and file bugs if/when the performance is bad
#-----

# Atlas based libraries
#ATLASHOME= $(HOME)/archives/BLAS_OPT/ATLAS/lib/Linux_ATHLONXP_SSE1/
#BLAS= -L$(ATLASHOME) -lf77blas -latlas

# use specific libraries (default library path points to other libraries)
#BLAS= $(ATLASHOME)/libf77blas.a $(ATLASHOME)/libatlas.a

# use the mkl Intel libraries for p4 (www.intel.com)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
#BLAS=-L/opt/intel/mkl/lib/32 -lmkl_p4 -lpthread

# LAPACK, simplest use vasp.5.lib/lapack_double
#LAPACK= ./vasp.5.lib/lapack_double.o

# use atlas optimized part of lapack
#LAPACK= ./vasp.5.lib/lapack_atlas.o -llapack -lcblas

# use the mkl Intel lapack
#LAPACK= -lmkl_lapack

# On Cray systems libsci with blas and lapack should be automagically linked
BLAS=
#LAPACK=
LAPACK= ./vasp.5.lib/lapack_double.o

#-----
#LIB = -L./vasp.5.lib -ldmy \
#      ./vasp.5.lib/linpack_double.o $(LAPACK) \
#      $(BLAS)

# options for linking (none required)
LINK = $(OFLAG)

#-----
# fft libraries:
# VASP.5.2 can use fftw.3.1.X (http://www.fftw.org)
# since this version is faster on P4 machines, we recommend to use it
#-----

#FFT3D = fft3dfurth.o fft3dlib.o

# alternatively: fftw.3.1.X is slightly faster and should be used if available
#FFT3D = fftw3d.o fft3dlib.o /opt/libs/fftw-3.1.2/lib/libfftw3.a

#=====
# MPI section, uncomment the following lines until
# general rules and compile lines
# presently we recommend OPENMPI, since it seems to offer better
# performance than lam or mpich
#
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
# !!! Please do not send me any queries on how to install MPI, I will
# certainly not answer them !!!!
#=====
#-----
# fortran linker for mpi
#-----

#FC=mpif77
#FCL=$ (FC)

#-----
# additional options for CPP in parallel version (see also above):
# NGZhalf          charge density reduced in Z direction
# wNGZhalf         gamma point only reduced in Z direction
# scalAPACK        use scalAPACK (usually slower on 100 Mbit Net)
#-----

CPP      = $(CPP_) -DMPI -DHOST=\"CrayXT/LinuxPgi\" \
           -Dkind8 -DCACHE_SIZE=8000 -DPGF90 \
           -DNGZhalf \
           -Daviodalloc \
           -DscalAPACK \
           -DRPROMU_DGEMV \
           -DMPI_BLOCK=4000000 -Duse_collective

#-----
# location of SCALAPACK
# if you do not use SCALAPACK simply leave that section commented out
#-----

#BLACS=/usr/local/BLACS_lam
#SCA_= /usr/local/SCALAPACK_lam

#SCA= $(SCA_)/scalapack_LINUX.a $(SCA_)/pblas_LINUX.a $(SCA_)/tools_LINUX.a \
# $(BLACS)/LIB/blacsF77init_MPI-LINUX-0.a $(BLACS)/LIB/blacs_MPI-LINUX-0.a $(BLACS)/LIB/blacsF77i

SCA=

#-----
# libraries for mpi
#-----
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
LIB      = -L../vasp.5.lib -ldmy \
           ../vasp.5.lib/linpack_double.o $(LAPACK) \
           $(SCA) $(BLAS) $(FFTW_LIB)

# FFT: fftmpi.o with fft3dlib of Juergen Furthmueller
#FFT3D   = fftmpi.o fftmpi_map.o fft3dfurth.o fft3dlib.o

# alternatively: fftw.3.1.X is slightly faster and should be used if available
#FFT3D   = fftmpi.o fftmpi_map.o fftw3d.o fft3dlib.o /opt/libs/fftw-3.1.2/lib/libfftw3.a
FFT3D   = fftmpiw.o fftmpi_map.o fftw3d.o fft3dlib.o

#-----
# general rules and compile lines
#-----

BASIC= symmetry.o symlib.o lattlib.o random.o

SOURCE= base.o      mpi.o      smart_allocate.o      xml.o \
        constant.o jacobi.o  main_mpi.o  scala.o \
        asa.o       lattice.o poscar.o  ini.o  mgrid.o  xclib.o  vdw_nl.o  xclib_grad.o \
        radial.o    pseudo.o  gridq.o   ebs.o \
        mkpoints.o wave.o   wave_mpi.o wave_high.o \
        $(BASIC)   nonl.o   nonlr.o  nonl_high.o dfast.o  choleski2.o \
        mix.o       hamil.o  xcgrad.o  xcspin.o potex1.o  potex2.o \
        constrmag.o cl_shift.o relativistic.o LDApU.o \
        paw_base.o metagga.o egrad.o   pawsym.o  pawfock.o  pawlfh.o  rhfatm.o  paw.o \
        mkpoints_full.o charge.o  Lebedev-Laikov.o stockholder.o dipol.o  pot.o \
        dos.o       elf.o    tet.o     tetweight.o hamil_rot.o \
        steep.o    chain.o  dyna.o    sphpro.o  us.o   core_rel.o \
        aedens.o   wavpre.o wavpre_noio.o broyden.o \
        dynbr.o    rmm-diis.o reader.o writer.o  tutor.o  xml_writer.o \
        brent.o    stufak.o fileio.o opergrid.o stepver.o \
        chgloc.o   fast_aug.o fock.o   mkpoints_change.o sym_grad.o \
        mymath.o   internals.o dynconstr.o dimer_heyden.o dvvtrajectory.o vdwforcefield.o \
        hamil_high.o nmr.o   pead.o    mlwf.o   subrot.o  subrot_scf.o \
        force.o    pwlfh.o  gw_model.o optreal.o davidson.o david_inner.o \
        electron.o rot.o   electron_all.o shm.o   pardens.o paircorrection.o \
        optics.o   constr_cell_relax.o stm.o   finite_diff.o elpol.o \
        hamil_lr.o rmm-diis_lr.o subrot_cluster.o subrot_lr.o \
        lr_helper.o hamil_lrf.o elinear_response.o ilinear_response.o \
        linear_optics.o linear_response.o \
        setlocalpp.o wannier.o electron_OEP.o electron_lhf.o twoelectron4o.o \
        ratpol.o   screened_2e.o wave_cacher.o chi_base.o wpot.o local_field.o \
        umps2.o    bse_te.o bse.o acfdt.o chi.o sydmat.o dmft.o
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
rmm-diis_mlr.o  linear_response_NMR.o

INC=

PAPI_OBJ=krp-rpt.o krp-rpt-init.o krp-rpt-init-sum.o krp-init-sum.c krp-init.o get-rnd.o

vasp.xt_vanilla_papi: $(SOURCE) $(FFT3D) $(INC) main_papi.o
rm -f vasp.xt_vanilla_papi
$(FCL) -o vasp.xt_vanilla_papi main_papi.o $(PAPI_OBJ) $(SOURCE) \\
$(FFT3D) $(LIB) $(LINK) $(PAPI_POST_LINK_OPTS)

makeparam: $(SOURCE) $(FFT3D) makeparam.o main.F $(INC)
$(FCL) -o makeparam $(LINK) makeparam.o $(SOURCE) $(FFT3D) $(LIB)
zgemmtest: zgemmtest.o base.o random.o $(INC)
$(FCL) -o zgemmtest $(LINK) zgemmtest.o random.o base.o $(LIB)
dgemmtest: dgemmtest.o base.o random.o $(INC)
$(FCL) -o dgemmtest $(LINK) dgemmtest.o random.o base.o $(LIB)
ffttest: base.o smart_allocate.o mpi.o mgrid.o random.o ffttest.o $(FFT3D) $(INC)
$(FCL) -o ffttest $(LINK) ffttest.o mpi.o mgrid.o random.o smart_allocate.o base.o $(FFT3D) $(LIB)
kpoints: $(SOURCE) $(FFT3D) makekpoints.o main.F $(INC)
$(FCL) -o kpoints $(LINK) makekpoints.o $(SOURCE) $(FFT3D) $(LIB)

clean:
-rm -f *.g *.f *.f90 *.o *.L *.mod *~; touch *.F

main_papi.o: main_papi$(SUFFIX)
cc -c get-rnd.c
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt.c
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt-init.c
cc -c ${PAPI_INCLUDE_OPTS} krp-rpt-init-sum.c
cc -c ${PAPI_INCLUDE_OPTS} krp-init-sum.c
cc -c ${PAPI_INCLUDE_OPTS} krp-init.c
$(FC) $(FFLAGS) $(DEBUG) $(INCS) -c main_papi$(SUFFIX)

xcgrad.o: xcgrad$(SUFFIX)
$(FC) $(FFLAGS) $(INLINE) $(INCS) -c xcgrad$(SUFFIX)
xcspin.o: xcspin$(SUFFIX)
$(FC) $(FFLAGS) $(INLINE) $(INCS) -c xcspin$(SUFFIX)

makeparam.o: makeparam$(SUFFIX)
$(FC) $(FFLAGS) $(DEBUG) $(INCS) -c makeparam$(SUFFIX)

makeparam$(SUFFIX): makeparam.F main.F
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
#  
# MIND: I do not have a full dependency list for the include  
# and MODULES: here are only the minimal basic dependencies  
# if one strucuture is changed then touch_dep must be called  
# with the corresponding name of the structure  
#  
base.o: base.inc base.F  
mgrid.o: mgrid.inc mgrid.F  
constant.o: constant.inc constant.F  
lattice.o: lattice.inc lattice.F  
setex.o: setexm.inc setex.F  
pseudo.o: pseudo.inc pseudo.F  
poscar.o: poscar.inc poscar.F  
mkpoints.o: mkpoints.inc mkpoints.F  
wave.o: wave.F  
nonl.o: nonl.inc nonl.F  
nonlr.o: nonlr.inc nonlr.F  
  
$(OBJ_HIGH):  
$(CPP)  
$(FC) $(FFLAGS) $(OFLAG_HIGH) $(INCS) -c $*$(SUFFIX)  
$(OBJ_NOOPT):  
$(CPP)  
$(FC) $(FFLAGS) $(INCS) -c $*$(SUFFIX)  
  
fft3dlib_f77.o: fft3dlib_f77.F  
$(CPP)  
$(F77) $(FFLAGS_F77) -c $*$(SUFFIX)  
  
.F.o:  
$(CPP)  
$(FC) $(FFLAGS) $(OFLAG) $(INCS) -c $*$(SUFFIX)  
.F$(SUFFIX):  
$(CPP)  
$(SUFFIX).o:  
$(FC) $(FFLAGS) $(OFLAG) $(INCS) -c $*$(SUFFIX)  
  
#  
# For safety we incorporate all the 4.6.* and 5.1.49 special rules historically developed  
# on XT using PGroup compilers. This is likely overkill
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
#  
  
# special rules  
#-----  
# these special rules are cummulative (that is once failed  
# in one compiler version, stays in the list forever)  
# -tpp5|6|7 P, PII-PIII, PIV  
# -xW use SIMD (does not pay off on PII, since fft3d uses double prec)  
# all other options do no affect the code performance since -O1 is used  
  
fft3dlib.o : fft3dlib.F  
$(CPP)  
# $(FC) -Mfree -O1 -tpp7 -xW -prefetch- -prev_div -unroll0 -vec_report3 -c $*$(SUFFIX)  
$(FC) -Mfree -O1 -unroll0 -c $*$(SUFFIX)  
  
fft3dfurth.o : fft3dfurth.F  
$(CPP)  
$(FC) -Mfree -O1 -c $*$(SUFFIX)  
  
radial.o : radial.F  
$(CPP)  
$(FC) -Mfree -O1 -c $*$(SUFFIX)  
  
symlib.o : symlib.F  
$(CPP)  
$(FC) -Mfree -O1 -c $*$(SUFFIX)  
  
symmetry.o : symmetry.F  
$(CPP)  
$(FC) -Mfree -O1 -c $*$(SUFFIX)  
  
wave_mpi.o : wave_mpi.F  
$(CPP)  
$(FC) -Mfree -O1 -c $*$(SUFFIX)  
  
wave.o : wave.F  
$(CPP)  
$(FC) -Mfree -O1 -c $*$(SUFFIX)  
  
dynbr.o : dynbr.F  
$(CPP)  
$(FC) -Mfree -O1 -c $*$(SUFFIX)
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
asa.o : asa.F
$(CPP)
$(FC) -Mfree -O1 -c $*$(SUFFIX)

broyden.o : broyden.F
$(CPP)
$(FC) -Mfree -O1 -c $*$(SUFFIX)
# Compiling with -O2 using PGI 10.3 will cause crash
# $(FC) -Mfree -O2 -c $*$(SUFFIX)

us.o : us.F
$(CPP)
$(FC) -Mfree -O1 -c $*$(SUFFIX)

LDApU.o : LDApU.F
$(CPP)
$(FC) -Mfree -O2 -c $*$(SUFFIX)
```

Q2 Input Settings

Inputs are given for four testcases, engine-8296, engine-11691, engine-12494, and engine-14636.

For Q2, the value of NPAR, which affects plane wave distribution, was chosen to be (no. cpus total/no. cpus per node), i.e. NPAR=1 for a 24 core run on hopper, NPAR=2 for a 48 core run, and NPAR=8 for a 96 core run on jaguar. This corresponds to a reasonable but not necessarily fully-optimized choice for each individual simulation. The initial value of NPAR=1 provided little scaling over one node.

Settings for “engine-8296” testcase POSCAR:

```
Poscar v1.1 :: 1 :: O K V Rb Tb
1.000000000000000
 6.0276281310899975    0.000000000000000    0.000000000000000
 -3.0138140655449988    5.2200790861084680    0.000000000000000
 0.000000000000000    0.000000000000000    7.8132793706162120
O   K   V   Rb   Tb
 8     2     2     1     1
Direct
 0.8245335689898001  0.1754664310101988  0.1730490409848015
 0.8245335689898010  0.6490671379795954  0.1730490409848015
 0.3509328620204055  0.1754664310101988  0.1730490409848015
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

0.6490671379795946	0.8245335689898010	0.8269509590151984
0.1754664310101988	0.3509328620204047	0.8269509590151984
0.1754664310101997	0.8245335689898010	0.8269509590151984
0.3333333332999970	0.6666666667000030	0.5355335322854755
0.6666666667000030	0.3333333332999970	0.4644664677145245
0.6666666667000030	0.3333333332999970	0.7999441298862570
0.3333333332999970	0.6666666667000030	0.2000558701137429
0.6666666667000030	0.3333333332999970	0.2502780209413001
0.3333333332999970	0.6666666667000030	0.7497219790586998
0.0000000000000000	0.0000000000000000	0.5000000000000000
0.0000000000000000	0.0000000000000000	0.0000000000000000

INCAR:

```
#Default RubyVasp from initialize
ISIF = 3
NSW = 200
NELM = 100
ISPIN = 2
ALGO = FAST
SYSTEM = RubyVasp :: O K V Rb Tb
ENCUT = 520
LREAL = AUTO
EDIFF = 0.0000020
PREC = Accurate
NELMIN = 3
ISMEAR = -5
SIGMA = 0.05
LWAVE = .TRUE.
ICHARG = 1
NPAR = 4
MAGMOM = 8*0.6 2*0.6 2*5.0 1*0.6 1*0.6
IBRION=1
```

KPOINTS:

```
#Generated by RubyVasp
0
Gamma
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

6 6 4
0 0 0

Settings for “engine-11691” testcase: POSCAR:

```
Poscar v1.1 :: 1 :: O Nd Hg
1.000000000000000
 3.7592158357167631    0.0000000000001069    0.8003035763575148
 1.6099791930998628    6.8918526554155637    1.6666618999953213
 -0.0008230340167181   0.0088086291353580    9.2329822558480572
O   Nd   Hg
 8     4     2
Direct
 0.2039622310617390  0.6500857749125036  0.9419897629640172
 0.7960377689382611  0.3499142250874964  0.0580102370359830
 0.0365484828224519  0.2058078366580189  0.7210951976970862
 0.9634515171775482  0.7941921633419810  0.2789048023029140
 0.4195392362750509  0.5071140658810024  0.6538074615688962
 0.5804607637249491  0.4928859341189974  0.3461925384311039
 0.2283310869105282  0.1538796049916262  0.3894582211873223
 0.7716689130894721  0.8461203950083738  0.6105417788126776
 0.6197484513759052  0.2202140142749508  0.5402890829732312
 0.3802515486240947  0.7797859857250492  0.4597109170267686
 0.8412991012578044  0.5189620633787531  0.7984397341056360
 0.1587008987421955  0.4810379366212472  0.2015602658943642
 0.1149651082363743  0.9314387909296988  0.8386309925975590
 0.8850348917636258  0.0685612090703012  0.1613690074024406
```

INCAR:

```
#Default RubyVasp from initialize
ISIF = 3
NSW = 200
NELM = 100
ISPIN = 2
ALGO = FAST
SYSTEM = RubyVaspy :: O Nd Hg
ENCUT = 520
LREAL = AUTO
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
EDIFF = 0.0000020
PREC = Accurate
NELMIN = 3
ISMEAR = -5
SIGMA = 0.05
LWAVE = .TRUE.
ICHARG = 1
NPAR = 4
MAGMOM = 8*0.6 4*0.6 2*5.0
IBRION=1
```

KPOINTS:

```
#Generated by RubyVasp
0
Monkhorst
8 4 4
0 0 0
```

Settings for “engine-12494” testcase: POSCAR:

```
Poscar v1.1 :: 1 :: O Nd Hg
1.000000000000000
 3.7592158357167631    0.0000000000001069    0.8003035763575148
 1.6099791930998628    6.8918526554155637    1.6666618999953213
 -0.0008230340167181    0.0088086291353580    9.2329822558480572
O   Nd   Hg
 8     4     2
Direct
 0.2039622310617390  0.6500857749125036  0.9419897629640172
 0.7960377689382611  0.3499142250874964  0.0580102370359830
 0.0365484828224519  0.2058078366580189  0.7210951976970862
 0.9634515171775482  0.7941921633419810  0.2789048023029140
 0.4195392362750509  0.5071140658810024  0.6538074615688962
 0.5804607637249491  0.4928859341189974  0.3461925384311039
 0.2283310869105282  0.1538796049916262  0.3894582211873223
 0.7716689130894721  0.8461203950083738  0.6105417788126776
 0.6197484513759052  0.2202140142749508  0.5402890829732312
 0.3802515486240947  0.7797859857250492  0.4597109170267686
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
0.8412991012578044 0.5189620633787531 0.7984397341056360
0.1587008987421955 0.4810379366212472 0.2015602658943642
0.1149651082363743 0.9314387909296988 0.8386309925975590
0.8850348917636258 0.0685612090703012 0.1613690074024406
```

INCAR:

```
#Default RubyVasp from initialize
ISIF = 3
NSW = 200
NELM = 100
ISPIN = 2
ALGO = FAST
SYSTEM = RubyVaspy :: O Nd Hg
ENCUT = 520
LREAL = AUTO
EDIFF = 0.0000020
PREC = Accurate
NELMIN = 3
ISMEAR = -5
SIGMA = 0.05
LWAVE = .TRUE.
ICHARG = 1
NPAR = 4
MAGMOM = 8*0.6 4*0.6 2*5.0
IBRION=1
```

KPOINTS:

```
#Generated by RubyVasp
0
Monkhorst
8 4 4
0 0 0
```

Settings for “engine-14636” testcase: POSCAR:

Poscar v1.1 :: 1 :: S Rb Nb Ag

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

1.000000000000000
5.8008929232369359 0.000000000003100 1.4615588754152482
2.9004464616182859 6.9805537426229014 0.7307794377602173
-0.0428748389150950 0.0000000000026181 12.4127526992230610

S Rb Nb Ag
8 4 2 2

Direct

0.5640493634484615	0.5530827124729448	0.6387117550532849
0.2558438309746911	0.9469172875270552	0.8612882449467151
0.8828679240785939	0.5530827124729450	0.8612882449467150
0.7972388814982536	0.9469172875270552	0.6387117550532850
0.4359506365515388	0.4469172875270551	0.3612882449467150
0.2027611185017462	0.0530827124729449	0.3612882449467151
0.1171320759214062	0.4469172875270552	0.1387117550532850
0.7441561690253091	0.0530827124729449	0.1387117550532850
0.5540193547450634	0.2500000000000000	0.8919612905098737
0.4459806452549369	0.7500000000000000	0.1080387094901264
0.1959806452549368	0.2500000000000000	0.6080387094901263
0.8040193547450634	0.7500000000000000	0.3919612905098737
0.6250000000000000	0.7500000000000000	0.7500000000000000
0.3750000000000000	0.2500000000000000	0.2500000000000000
0.1250000000000000	0.7500000000000000	0.7500000000000000
0.8750000000000000	0.2500000000000000	0.2500000000000000

INCAR:

```
#Default RubyVasp from initialize
ISIF = 3
NSW = 200
NELM = 100
ISPIN = 2
ALGO = FAST
SYSTEM = RubyVasp :: S Rb Nb Ag
ENCUT = 520
LREAL = AUTO
EDIFF = 0.0000020
PREC = Accurate
NELMIN = 3
ISMEAR = -5
SIGMA = 0.05
LWAVE = .TRUE.
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
ICHARG = 1
NPAR = 4
MAGMOM = 8*0.6 4*0.6 2*5.0 2*5.0
IBRION=1
```

KPOINTS:

```
#Generated by RubyVasp
0
Monkhorst
6 6 4
0 0 0
```

Q2 Run Script The batch scripts copy the necessary files to the scratch directory, run the simulation, and copy the key output files back the submission directory.

The file given below is for a 96 core run on jaguar. Submissions on hopper used an identical script except for using a differently named scratch directory.

```
#!/bin/bash
#PBS -N OMB_Metric
#PBS -j oe
#PBS -M kentpr@ornl.gov
#PBS -m abe
#PBS -A csc091qmcpac
#PBS -l walltime=2:00:00,size=96
#PBS -l gres=widow3

ENDSUF=a
NCORES=96

export NO_STOP_MESSAGE=1

VASP_BINARY=./vasp.xt_vanilla_papi

SUFFIX=_${NCORES}${ENDSUF}

cd ${PBS_O_WORKDIR}
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
if [ ! -e COMPLETE ]; then

if [ -e LASTOK ]; then

rm -f LASTOK

echo >_sed.scr s/.*/${USER}_/g
wdir='pwd | sed 's///_/' | sed -f _sed.scr'
rm -f _sed.scr
SCRATCH=/tmp/work/pk7
tmpdir=${SCRATCH}/${wdir}${SUFFIX}
if [ ! -e $tmpdir ] ; then
mkdir $tmpdir
fi
echo Using $tmpdir

#
# Tidy previous run
#

for f in CONTCAR${SUFFIX} OSZICAR${SUFFIX} OUTCAR${SUFFIX} XDATCAR${SUFFIX}
do

if [ -e ${f}.bz2 ]; then

bunzip2 -f ${f}.bz2

fi

done

if [ -e OUTCAR${SUFFIX} ]; then

if [ ! -e XDATCAR${SUFFIX} ]; then

echo No XDATCAR${SUFFIX} present. Did the job run? add some logic - perhaps on /scratch?
echo ABORT
exit 1

fi
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
n=1

#while [ -e OUTCAR${SUFFIX}_${n} || -e OUTCAR${SUFFIX}_${n}.bz2 ]; do
while [ -e OUTCAR${SUFFIX}_${n} ]; do
    echo Found OUTCAR${SUFFIX}_${n}
    n=`expr $n + 1`
done
echo New files renumbered _${n}

for f in OUTCAR${SUFFIX} OSZICAR${SUFFIX} CONTCAR${SUFFIX} XDATCAR${SUFFIX} POSCAR
do
if [ -e ${f} ]; then
    chmod og+r ${f}
    mv ${f} ${f}__${n}
    ls -l ${f}__${n}
fi

done

if [ -e CONTCAR${SUFFIX}_${n} ]; then
    cp -p CONTCAR${SUFFIX}_${n} POSCAR
fi

fi

#
# Setup new actual run
#

for f in POSCAR POTCAR KPOINTS
do
    cp -f ${f} $tmpdir
done

rm -f $tmpdir/INCAR
if [ -e INCAR${SUFFIX} ]; then
    cp INCAR${SUFFIX} $tmpdir/INCAR
    echo Using INCAR${SUFFIX}
else
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
echo Using INCAR
cp INCAR $tmpdir
fi

cd $tmpdir
pwd
ls -l INCAR
cat INCAR
echo --- ls -l
ls -l
echo --- Using executable $VASP_BINARY
echo `ls -l $VASP_BINARY'
rm -f vasp; cp -p $VASP_BINARY vasp
echo --- Start `date'
time aprun -n $NCORES ./vasp
echo --- End `date'
cat INCAR >>OUTCAR
echo Run in $tmpdir >>OUTCAR
ls -l >>OUTCAR

for f in OSZICAR OUTCAR CONTCAR XDATCAR
do

if [ -e $PBS_O_WORKDIR/NO_BZIP2 ]; then
  cp -f -p ${f} $PBS_O_WORKDIR/${f}${SUFFIX}
else
  bzip2 -f ${f}
  cp -f -p ${f}.bz2 $PBS_O_WORKDIR/${f}${SUFFIX}.bz2
fi

done

echo >$PBS_O_WORKDIR/LASTOK

fi

fi
```

0.7.3 NIMROD

Benchmark Problem 1

Q2 Modules *hopper.nersc.gov*

Currently Loaded Modulefiles:

- 1) modules/3.2.6.6
- 2) xtpe-network-gemini
- 3) pgi/11.9.0
- 4) xt-libsci/11.0.05
- 5) udreg/2.3.1-1.0400.3911.5.13.gem
- 6) ugni/2.3-1.0400.4127.5.20.gem
- 7) pmi/3.0.0-1.0000.8661.28.2807.gem
- 8) dmapp/3.2.1-1.0400.3965.10.63.gem
- 9) gni-headers/2.1-1.0400.4156.6.1.gem
- 10) xpmem/0.1-2.0400.30792.5.6.gem
- 11) xe-sysroot/4.0.36
- 12) xt-asyncpe/5.07
- 13) atp/1.4.2
- 14) PrgEnv-pgi/4.0.36
- 15) eswrap/1.0.10
- 16) xtpe-mc12
- 17) xt-shmem/5.4.4
- 18) xt-mpich2/5.4.4
- 19) torque/2.5.9
- 20) moab/6.1.4
- 21) acml/4.4.0
- 22) superlu_dist/2.5_g
- 23) superlu/4.0_g
- 24) parmetis/3.1.1_g
- 25) papi/4.2.0

Q2 Compilation

File make.inc for the Linux operating system on the Cray XT:

This file contains machine-specific definitions:

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
# 0) Home directory locations--done first to avoid repetition.

# 1) The first group of definitions determines the compilers and

# compiler options.

# 2) The machine-dependent 'local' file to use in nimlib.

# 3) The next group determines whether mpi is used in order to run

# the physics kernel in parallel.

# 4) The third group determines which mathematical libraries are linked

# and, hence, which linear solver options may be used.

# PART 0: HOME LOCATIONS FOR LIBRARIES, ETC.

# See the respective library definitions in parts 3 and 4.

BLAS_HOME =

MPI_HOME =

SLU_HOME = /usr/common/acts/SuperLU/SuperLU_DIST/2.5

# PART 1: GENERAL COMPILER DEFINITIONS

# General fortran 90 compiler and loader definitions:

#Portland Group compiler

FFLAGS = -fastsse -byteswapio -r8 -fPIC

FFLAGS_DBG = -g -C -r8 -Mbounds -byteswapio

FFLAGS_NODBL = -fast -byteswapio
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
FFLAGS_NOOPT = -byteswapio -r8

F90 = ftn -target=linux

MPIF90 = $(F90)

# Fortran loader:

LDR_FLAGS =

LDR = $(F90)

MPILDR = $(MPIF90)

# General C compiler definitions:

CFLAGS =

CFLAGS_DBG = -g -O2

CC = cc -target=linux

MPICC = cc

#Archive utilities.

# AR creates the archive and RANLIB creates the index

AR = ar -r

RANLIB = ranlib

# PART 2: MACHINE-DEPENDENT LOCAL FILE IN NIMLIB

LOCAL_FILE = local_xt.f

# PART 3: MPI DEFINITIONS
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
# MPI library and include paths need to be set when creating a

# executable for parallel computation. First, comment-out the

# mpi_serial.f link, and uncomment the mpi_parallel.f link. Then,

# set the mpi library and include paths and library path names.

# These names may not need to be defined when the compiler is

# an mpi* version of the F90 compiler. The default definitions

# create a serial version of NIMROD.

#MPILINK = mpi_serial.f

MPILINK = mpi_parallel.f

MPI_LIB =

MPI_INCL =

# PART 4: MATH LIBRARY DEFINITIONS

# Links to an appropriate math library are needed to run the NIMROD

# physics kernel using direct solves for the linear systems, i.e.

# setting the namelist input item solver to any of the following:

# 1) 'lapack' -- banded serial LAPACK solves

# 2) 'seq_slu' -- sparse serial solves via Sequential SuperLU

# 3) 'slu_dist' -- sparse parallel solves via SuperLU_DIST

# [SuperLU_DIST is our present recommendation for large parallel
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
# simulations. The SuperLU packages were written by Xiaoye Li (LBL),  
# James Demmel (UC-Berkeley), and John Gilbert (UC-SB). To find  
# documentation, see the ACTS Collection web page at NERSC,  
# http://acts.nersc.gov/superlu/]  
  
# To use one of the solver libraries, one needs to:  
  
# A. Specify a BLAS library.  
  
# B. Set definitions for compiling the C bridge routines.  
  
# C. Set library names for the math libraries.  
  
# Dummy external links are also available for compiling the kernel  
# without any external libraries (but don't expect to use the  
# solver options listed above). They are also used for NIMPLLOT.  
  
# Leave all of the following variables blank to use the dummy  
# externals in nimrod.  
  
# A. BLAS library:  
  
# Set the full BLAS library path name in BLAS_LIB.  
  
BLAS_LIB =  
  
# B. Specify definitions for the C bridge routines:  
  
# Sequential SuperLU and SuperLU_DIST are written in C, and  
# there are 'bridge' routines written in C for calling these
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
# libraries. NIMROD only calls these bridge libraries (or their
# dummy stubs). To compile a set of actual bridge

# routines, uncomment the respective SLU_LINK variable below, set the

# compiler options for the bridge routines, and set the source paths

# for the libraries so that the C compiler can

# find library-specific include files. There may be conflicts

# in the include path files if one tries to link both SuperLU

# libraries.

SLU_SRC = $(SLU_HOME)/SRC

#SLU_LINK = sslu_link

SLU_LINK = slud_link

CFLAGS_SLU = -O3

CFLAGS_SLU_DBG = $(CFLAGS_DBG)

CINCL_SLU = -I$(SLU_SRC)

# C. Set the full path names for the math libraries:

# Leaving the paths as $(NIMHOME)/externals/lib*_dummy.a loads the

# physics kernel with the nonfunctioning dummy external for the respective

# library. Include the SLU_LINK library with one of the functioning

# SuperLU libraries. You will be able to the actual LAPACK library and/or
```

```
# one of the actual SuperLU libraries.

# The new version of SuperLU_DIST (2.1) requires ParMETIS to be

# loaded, and the METIS_LIB variable helps with this. If you use an older

# version or the sequential SuperLU, METIS_LIB can be blank.

LAP_LIB = $(NIMHOME)/externals/liblapack_dummy.a

SSLU_LIB = $(NIMHOME)/externals/libssl_dummy.a

SLUD_LIB = $(NIMHOME)/externals/lib$(SLU_LINK).a \
$(SLU_HOME)/craypgi-xe6_g/lib/libsuperlu_dist_2.5.a

METIS_LIB = /usr/common/acts/PARMETIS/3.1.1/craypgi-xe6_g/libparmetis.a \
/usr/common/acts/PARMETIS/3.1.1/craypgi-xe6_g/libmetis.a \
$(NIMHOME)/krp/libkrplib.a $(PAPI_POST_LINK_OPTS) -lm
```

Q2 Input Settings nimrod.in : *medium linear case*
&grid_input

```
gridshape = 'flux'
```

```
geom = 'tor'
```

```
mx = 40
```

```
my = 65
```

```
poly_degree = 5
```

```
nxbl = 4
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
nybl =6

lin_nmodes = 1

lin_nmax = 6

nlayers = 1

lphi = 6 ! 2 modes

/

&equil_input

ds_use=elecd

dvac = 1.e0

dexp = 500.

xvac = 0.751

tor_eqja_fe=.true.

beta=0.1

/

&physics_input

continuity='fix profile'

nonlinear = .false.

ohms = 'mhd'

advect = 'V only'
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
eta_model=fixed

elecd = 1.931E-02 ! S=3.802e8

kin_visc = 1.00E+00 ! Pr=16.73

nd_diff=0.000

/

&closure_input

p_model = 'anisol'

k_perp = 1.00e-00

k_pll = 1.E6

/

&numerical_input

dtm = 5.0E-8 ! Tau_A = s

tmax = 120.

cpu_tmax = 1650

nstep = 100

ngr = 2

si_fac_mhd=1.5

si_fac_pres=1.5

si_fac_j0=1.0
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
si_fac_nl=1.5

mhd_si_iso=0.E+0

v_cfl=0.25

fv_vdgv=0.51

fb_vxb=0.51

fp_vdgp=0.51

dt_incr=1.04

dt_change_frac=0.05

ave_change_limit=3.e-3

divbd = 1.5e5

kdivb_2_limit = 10000.

/

&solver_input

solver = 'slu_dstm'

off_diag_fac = 0.9

maxit = 80

tol = 1.e-9

extrap_order = 2

/
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
&output_input

dump_file = 'dump.00000'

dump_name = 'dump'

dump_dir = '.'

ndump = 2000

dump_over = 0

nhist = 5

hist_binary = T

ihist = 2

jhist = 2

xdraw_dir = '.'

xy_stride = 0

xt_stride = 0

yt_stride = 0

x0fac = 0.5

y0fac = 0.125

ndxout = 0

dx_dir = '.'

detflag = F
```

itflag = F

Q2 Input Settings nimrod.in : *large linear case*

&grid_input

gridshape = 'flux'

geom = 'tor'

mx = 80

my = 128

poly_degree = 5

nxbl =8

nybl =16

lin_nmodes = 1

lin_nmax = 6

nlayers = 1

lphi = 6

/

&equil_input

ds_use=elecd

dvac = 1.e0

dexp = 500.

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
xvac = 0.751

tor_eqja_fe=.true.

beta=0.1

/

&physics_input

continuity='fix profile'

nonlinear = .false.

ohms = 'mhd'

advect = 'V only'

eta_model=fixed

elecd = 1.931E-02 ! S=3.802e8

kin_visc = 1.00E+00 ! Pr=16.73

nd_diff=0.000

/

&closure_input

p_model = 'anisol'

k_perp = 1.00e-00

k_pll = 1.E6

/
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

&numerical_input

dtm = 5.0E-8 ! Tau_A = s

tmax = 120.

cpu_tmax = 1650

nstep = 100

ngr = 2

si_fac_mhd=1.5

si_fac_pres=1.5

si_fac_j0=1.0

si_fac_nl=1.5

mhd_si_iso=0.E+0

v_cfl=0.25

fv_vdgv=0.51

fb_vxb=0.51

fp_vdgp=0.51

dt_incr=1.04

dt_change_frac=0.05

ave_change_limit=3.e-3

divbd = 1.5e5

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
kdivb_2_limit = 10000.
```

```
/
```

```
&solver_input
```

```
solver = 'slu_dstm'
```

```
off_diag_fac = 0.9
```

```
maxit = 80
```

```
tol = 1.e-9
```

```
extrap_order = 2
```

```
/
```

```
&output_input
```

```
dump_file = 'dump.00000'
```

```
dump_name = 'dump'
```

```
dump_dir = '.'
```

```
ndump = 2000
```

```
dump_over = 0
```

```
nhist = 5
```

```
hist_binary = T
```

```
ihist = 2
```

```
jhist = 2
```

```
xdraw_dir = '.'
```

```
xy_stride = 0
```

```
xt_stride = 0
```

```
yt_stride = 0
```

```
x0fac = 0.5
```

```
y0fac = 0.125
```

```
ndxout = 0
```

```
dx_dir = '.'
```

```
detflag = F
```

```
itflag = F
```

Q2 Input Settings nimrod.in : *small nonlinear case*

```
&grid_input
```

```
gridshape = 'flux'
```

```
geom = 'tor'
```

```
mx = 40
```

```
my = 65
```

```
poly_degree = 5
```

```
nxbl = 4
```

```
nybl = 6
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
lin_nmodes = 11

lin_nmax = 10

nlayers = 6

lphi = 5 ! 11 modes

/

&equil_input

ds_use=none

dvac = 1.e0

dexp = 500.

xvac = 0.751

tor_eqja_fe=.true.

beta=0.1

/

&physics_input

continuity='fix profile'

nonlinear = .true.

ohms = 'mhd'

advect = 'V only'

eta_model=eta full
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
eta_ref_t=1669.

elecd_max=1.8

elecd = 1.931E-02 ! S=3.802e8

kin_visc = 1.00E+00 ! Pr=16.73

nd_diff=0.000

/

&closure_input

p_model = 'anisol'

k_perp = 1.00e-00

k_pll = 1.E6

/

&numerical_input

dtm = 5.0E-8 ! Tau_A = s

tmax = 120.

cpu_tmax = 15000

nstep = 1694

ngr = 2

si_fac_mhd=1.5

si_fac_pres=1.5
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
si_fac_j0=1.0
si_fac_nl=1.5
mhd_si_iso=0.E+0
v_cfl=0.25
fv_vdgv=0.51
fb_vxb=0.51
fp_vdgp=0.51
dt_incr=1.04
dt_change_frac=0.05
ave_change_limit=3.e-3
divbd = 1.5e5
kdivb_2_limit = 10000.
/
&solver_input
solver = 'slu_dstm'
off_diag_fac = 0.9
maxit = 80
tol = 1.e-9
extrap_order = 2
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
/  
  
&output_input  
  
dump_file = 'dump.00000'  
  
dump_name = 'dump'  
  
dump_dir = '.'  
  
ndump = 500  
  
dump_over = 0  
  
nhist = 5  
  
hist_binary = T  
  
ihist = 2  
  
jhist = 2  
  
xdraw_dir = '.'  
  
xy_stride = 0  
  
xt_stride = 0  
  
yt_stride = 0  
  
x0fac = 0.5  
  
y0fac = 0.125  
  
ndxout = 0  
  
dx_dir = '.'
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
detflag = F
```

```
itflag = F
```

Q2 Run Script nimrod.in : *medium linear case*

```
#PBS -q debug
```

```
#PBS -l mppwidth=24
```

```
#PBS -l mpnnppn=24
```

```
#PBS -l walltime=0:30:00
```

```
#PBS -N mhd_det
```

```
#PBS -A mp200
```

```
cd $PBS_O_WORKDIR
```

```
time aprun -n 24 -N 24 $SCRATCH/nimuw341/nimrod/nimrod
```

Q2 Run Script nimrod.in : *large linear case*

```
#PBS -q debug
```

```
#PBS -l mppwidth=128
```

```
#PBS -l mpnnppn=16
```

```
#PBS -l walltime=0:30:00
```

```
#PBS -N mhd_det
```

```
#PBS -A mp200
```

```
cd $PBS_O_WORKDIR
```

```
time aprun -n 128 -N 16 $SCRATCH/nimuw341/nimrod/nimrod
```

Q2 Run Script nimrod.in : *small nonlinear case*

```
#PBS -q regular

#PBS -l mppwidth=144

#PBS -l mppnppn=24

#PBS -l walltime=5:00:00

#PBS -N mhd_nl

#PBS -A mp200

cd $PBS_O_WORKDIR

time aprun -n 144 -N 24 $SCRATCH/nimuw341/nimrod/nimrod
```

Benchmark Problem 2

Q2 Modules

Currently Loaded Modulefiles:

- 1) modules/3.2.6.6
- 2) xtpe-network-gemini
- 3) pgi/11.9.0
- 4) xt-libsci/11.0.05
- 5) udreg/2.3.1-1.0400.3911.5.13.gem
- 6) ugni/2.3-1.0400.4127.5.20.gem
- 7) pmi/3.0.0-1.0000.8661.28.2807.gem
- 8) dmapp/3.2.1-1.0400.3965.10.63.gem
- 9) gni-headers/2.1-1.0400.4156.6.1.gem
- 10) xpmem/0.1-2.0400.30792.5.6.gem
- 11) xe-sysroot/4.0.36
- 12) xt-asyncpe/5.07
- 13) atp/1.4.2
- 14) PrgEnv-pgi/4.0.36
- 15) eswrap/1.0.10

```
16) xtpe-mc12
17) xt-shmem/5.4.4
18) xt-mpich2/5.4.4
19) torque/2.5.9
20) moab/6.1.4
21) vim/7.1
22) parmetis/3.1.1_g
23) subversion/1.6.4
24) java/jdk1.7.0_02
25) hpctoolkit
26) papi/4.2.0
27) tau/2.21.2
```

Q2 Compilation

```
>ftn -fast -Kieee
```

Q2 Input Settings

```
&grid_input
    gridshape='flux'
    eqfile='fluxgrid.dat'
    geom='tor'
    mx=64
    my=64
    nxbl=16
    nybl=16
    xmin=1.06
    xmax=2.26
    ymin=-1.0
    ymax=1.0
    xo=1.762
    yo=-2.486e-3
    vac_frac=.0625
    lphi=1
    lin_nmodes=1
    lin_nmax=1
```

```
    nlayers=1
/
&equil_input
    ds_function = 'spitzer'
/
&physics_input
    init_type="shear alf"
    eq_flow='none'
    advect='V only'
    nx=2
    ny=4
    bamp=1.e-10
    ndens=6.00e+19
    elecd=0.03632
    eta_ref_t=-1
    kin_visc=3.632
    dvac=1.0e4
    dexp=0.
    ds_use='both'
    ohms='mhd'
    nonlinear=.false.
    beta=1.
    continuity='fix profile'
    zero_bnorm=.true.
    be0=1.0
/
&closure_input
    p_model='adiabat'
    k_perp=1.
    k_pll=1.e8
/
&numerical_input
    poly_degree=4
    dtm=1.e-7
    v_cfl=0.5
    fv_vdgv=0.51
    fp_vdgp=0.51
    fb_vxb=0.51
    feta=1.
    fvsc=1.
    split_visc=.false.
    si_fac_mhd=1.5
    si_fac_pres=1.5
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
mhd_si_iso=0.  
si_fac_j0=1.0  
si_fac_nl=1.5  
split_divb=.false.  
fdivb=1.  
divbd=1.e5  
kdivb_2_limit=10.0  
ngr=2  
met_spl='iso'  
p_computation="at nodes"  
tmax=1.1e-1  
cpu_tmax=8e8  
nstep=50 /  
&solver_input  
    solver='slu_dist'  
    tol=1.e-10  
    extrap_order=2  
    maxit=100 /  
&output_input  
    n_probe=1  
    nhist=1  
    ihist=        40  
    jhist=         3  
    hist_binary=T  
    ndump=100  
    dump_over = 0  
    dump_file = 'dump.00000' /  
&particle_input  
    nm=32000000  
    restart=0  
    orbits=1  
    phqty=6  
    anisop=1  
    vhmx=2.23607e6  
    ecrit=10.  
    avu0=0.  
    avw0=0.05 , 0  
    rh0=1.8  
    mass0=3.3435860e-27  
    charge0=1.60217733e-19  
    R0=1.751  
    betafrac=0.16  
    Ppsi0=0.1322
```

```
part_type='drift'
psipmax=0.2644
tpartdr='.'
phdump=.true.
add_phot=1
bmxs=0
emxs=8
pnorm_field="pres_eq"
hot_distro="slowing down"
scale4phot=.true.
eparallel=0.
iseed=3
dump_part=.false.
/

```

Q2 Run Script

```
#PBS -A CSC091NIMROD
#PBS -l size=256
#PBS -l walltime=00:30:00
#PBS -l gres=widow1%widow2%widow3
#PBS -j eo

echo $PBS_O_WORKDIR
echo $HOME
cd $PBS_O_WORKDIR
pwd
time aprun -n 256 ./nimrod
```

0.7.4 QMCPACK

graphite-benchmark directory contains

- qmc.xml
- lda.pwscf.h5
- gr3x3x1.wfs.xml

- gr4x4x1.wfs.xml
- C.BFD.xml
- titan.pbs

Other target simulations will be structured similarly to this benchmark.

To streamline benchmark processes, the input templates assume that the common files, C.BFD.xml and lda.pwscf.h5 are located in the parent directory where QMCPACK is executed. The batch script, titan.pbs, creates a directory with the timestamp at the run time, copies input files to the run directory and run QMCPACK binary qmcapp. **Q2 Modules -CPUs**

- 1) modules/3.2.6.6
- 2) xe-sysroot/4.0.30.securitypatch.20110928
- 3) xtpe-network-gemini
- 4) PrgEnv-gnu/4.0.30
- 5) xt-mpich2/5.4.1
- 6) atp/1.4.1
- 7) xt-asyncpe/5.05
- 8) xe-sysroot/4.0.30
- 9) xpmem/0.1-2.0400.29883.4.6.gem
- 10) gni-headers/2.1-1.0400.3906.5.1.gem
- 11) dmapp/3.2.1-1.0400.3965.10.12.gem
- 12) pmi/3.0.0-1.0000.8661.28.2807.gem
- 13) ugni/2.3-1.0400.3912.4.29.gem
- 14) udreg/2.3.1-1.0400.3911.5.6.gem
- 15) xt-libsci/11.0.04.4
- 16) gcc/4.6.2
- 17) xtpe-interlagos
- 18) eswrap/1.0.9
- 19) lustredu/1.0
- 20) DefApps
- 21) altd/1.0
- 22) hdf5/1.8.7
- 23) fftw/3.3.0.0
- 24) cmake/2.8.6
- 25) subversion/1.6.17

Q2 Modules -GPUs

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
1) modules/3.2.6.6
2) xe-sysroot/4.0.30.securitypatch.20110928
3) xtpe-network-gemini
4) PrgEnv-gnu/4.0.30
5) xt-mpich2/5.4.1
6) atp/1.4.1
7) xt-asyncpe/5.05
8) xe-sysroot/4.0.30
9) xpmem/0.1-2.0400.29883.4.6.gem
10) gni-headers/2.1-1.0400.3906.5.1.gem
11) dmapp/3.2.1-1.0400.3965.10.12.gem
12) pmi/3.0.0-1.0000.8661.28.2807.gem
13) ugni/2.3-1.0400.3912.4.29.gem
14) udreg/2.3.1-1.0400.3911.5.6.gem
15) xt-libsci/11.0.04.4
16) gcc/4.6.2
17) xtpe-interlagos
18) eswrap/1.0.9
19) lustredu/1.0
20) DefApps
21) altd/1.0
22) hdf5/1.8.7
23) fftw/3.3.0.0
24) cmake/2.8.6
25) subversion/1.6.17
26) libsci_acc/1.0.02
27) cudatools/4.0.17a
28) cuda/4.0.17a
29) craype-accel-nvidia20
```

Q2 Compilation -CPUs

TitanGNU.cmake

```
SET(CMAKE_SYSTEM_PROCESSOR "XK6")

set(CMAKE_C_COMPILER /opt/cray/xt-asyncpe/5.05/bin/cc)
set(CMAKE_CXX_COMPILER /opt/cray/xt-asyncpe/5.05/bin/CC)
set(GNU_OPTS "-DADD_ -DINLINE_ALL=inline")
set(GNU_FLAGS "-fopenmp -O3 -Drestrict=__restrict__ -finline-limit=1000
               -fstrict-aliasing -funroll-all-loops -Wno-deprecated ")
```

```
set(XT_FLAGS "-march=bdver1 -msse3 -D_CRAYMPI")
set(CMAKE_CXX_FLAGS "${XT_FLAGS} ${GNU_FLAGS} -ftemplate-depth-60 ${GNU_OPTS}")
set(CMAKE_C_FLAGS "${XT_FLAGS} ${GNU_FLAGS} -std=c99")

SET(QMC_BUILD_STATIC 1)
SET(ENABLE_OPENMP 1)
SET(HAVE_MPI 1)
SET(HAVE_SSE 1)
SET(HAVE_SSE2 1)
SET(HAVE_SSE3 1)
SET(HAVE_SSSE3 1)
SET(USE_PREFETCH 1)
SET(PREFETCH_AHEAD 12)

set(CMAKE_FIND_ROOT_PATH_MODE_PROGRAM NEVER)
set(CMAKE_FIND_ROOT_PATH_MODE_LIBRARY ONLY)
set(CMAKE_FIND_ROOT_PATH_MODE_INCLUDE ONLY)
set(CMAKE_SHARED_LINKER_FLAGS "")

FOREACH(type SHARED_LIBRARY SHARED_MODULE EXE)
    SET(CMAKE_${type}_LINK_STATIC_C_FLAGS "-Wl,-Bstatic")
    SET(CMAKE_${type}_LINK_DYNAMIC_C_FLAGS "-static")
    SET(CMAKE_${type}_LINK_STATIC_CXX_FLAGS "-Wl,-Bstatic")
    SET(CMAKE_${type}_LINK_DYNAMIC_CXX_FLAGS "-static")
ENDFOREACH(type)

set(CMAKE_FIND_ROOT_PATH
    /opt/cray/hdf5/1.8.6-gnu/46
    /opt/fftw/3.3.0.0/interlagos
    /sw/xk6/boost/1.44.0/cle4.0_gnu4.5.3
    /ccs/proj/mat034/jnkim/xk6/gnu45/libxml2
)

set(EINSPLINE_SSE_BUG 1)
set(HAVE_EINSPLINE 1)
set(HAVE_EINSPLINE_EXT 0)
```

Q2 Compilation -GPUs

TitanCUDA.cmake

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
SET(CMAKE_SYSTEM_PROCESSOR "XK6")
#2012-02-17
#NEED THESESES + defaults
# module swap PrgEnv-pgi PrgEnv-gnu
# module load xtpe-accel-nvidia20
# module load cudatools
set(CMAKE_C_COMPILER /opt/cray/xt-asyncpe/5.05/bin/cc)
set(CMAKE_CXX_COMPILER /opt/cray/xt-asyncpe/5.05/bin/CC)
set(GNU_OPTS "-DADD_ -DINLINE_ALL=inline")
set(GNU_FLAGS " -fomit-frame-pointer -malign-double -fopenmp -O3
-Drestrict=__restrict__ -finline-limit=1000 -fstrict-aliasing
-funroll-all-loops ")
set(XT_FLAGS " -msse -msse2 -msse3 -D_CRAYMPI")
set(CMAKE_CXX_FLAGS "${XT_FLAGS} ${GNU_FLAGS} -ftemplate-depth-60
${GNU_OPTS} -Wno-deprecated ")
set(CMAKE_C_FLAGS "${XT_FLAGS} ${GNU_FLAGS} -std=c99")

# need for both c++ and c
set(QMC_CUDA 1)
SET(ENABLE_OPENMP 1)
SET(HAVE_MPI 1)
set(HAVE_CUDA 1)
SET(QMC_BUILD_STATIC 1)

SET(HAVE_SSE 1)
SET(HAVE_SSE2 1)
SET(HAVE_SSE3 1)
SET(HAVE_SSSE3 1)
SET(USE_PREFETCH 1)
SET(PREFETCH_AHEAD 12)

set(CMAKE_FIND_ROOT_PATH_MODE_PROGRAM NEVER)
set(CMAKE_FIND_ROOT_PATH_MODE_LIBRARY ONLY)
set(CMAKE_FIND_ROOT_PATH_MODE_INCLUDE ONLY)
set(CMAKE_SHARED_LINKER_FLAGS "")

set(CMAKE_FIND_ROOT_PATH
/opt/cray/hdf5/1.8.6/gnu/46
/opt/fftw/3.3.0.0/interlagos
/sw/xk6/boost/1.44.0/cle4.0_gnu4.5.3
```

```

/ccs/home/jnkim/xk6-gnu45/libxml2
)

set(HAVE_EINSPLINE 1)
set(HAVE_EINSPLINE_EXT 0)

include_directories(/opt/nvidia/cuda/4.0.17a/include )
set(CUDA_NVCC_FLAGS "-arch=sm_20;-Drestrict=__restrict__;
-DNO_CUDA_MAIN;-O3;-use_fast_math")
set(CUDA_CUDART_LIBRARY /opt/cray/nvidia/default/lib64/libcuda.so)
set(CUDA_CUDA_LIBRARY /opt/cray/nvidia/default/lib64/libcuda.so)
set(CUDA_TOOLKIT_ROOT_DIR /opt/nvidia/cudatools/4.0.17a)
set(CUDA_TOOLKIT_INCLUDE /opt/nvidia/cudatools/4.0.17a/include)
set(CUDA_LIBRARIES ${CUDA_CUDART_LIBRARY})
set(CUDA_make2cmake ${CMAKE_ROOT}/Modules/FindCUDA/make2cmake.cmake)
set(CUDA_parse_cubin ${CMAKE_ROOT}/Modules/FindCUDA/parse_cubin.cmake)
set(CUDA_run_nvcc ${CMAKE_ROOT}/Modules/FindCUDA/run_nvcc.cmake)
set(CUDA_PROPAGATE_HOST_FLAGS OFF) #do not propagate the host flags

```

Q2 Input Settings

This benchmark problem performs one VMC section and DMC section. The VMC section is used to generate uncorrelated configurations to perform the main DMC section. The input file contains the text fields (all in CAPTIAL letters) that must be replaced by other scripts to perform a series of runs.

- qmc.xml: main XML input file. [language=XML,keywords=qmc]

```

<simulation>
  <project id="TITLE" series="0"/>
  <random seed="SEED"/>
  <include href="WFSXML"/>
  <hamiltonian name="h0" type="generic" target="e">
    <pairpot name="ElecElec" type="coulomb" source="e" target="e"/>
    <pairpot name="IonIon" type="coulomb" source="ion0" target="ion0"/>
    <pairpot type="pseudo" name="PseudoPot" source="ion0"
      wavefunction="psi0" format="xml">
      <pseudo elementType="C" href="..../C.BFD.xml" format="xml"/>
    </pairpot>
  </hamiltonian>
</simulation>

```

```

</hamiltonian>
<init source="ion0"/>
<qmc method="vmc" checkpoint="100" move="pbyp" gpu="yes">
    <estimator name="LocalEnergy" hdf5="no"/>
    <parameter name="useDrift">yes</parameter>
    <parameter name="blocks">VMCBLOCKS</parameter>
    <parameter name="steps">VMCSTEPS</parameter>
    <parameter name="walkers">WALKERS</parameter>
    <parameter name="samples">SAMPLES</parameter>
    <parameter name="warmupsteps">10</parameter>
    <parameter name="timestep">2.0</parameter>
</qmc>
<qmc method="dmc" checkpoint="100" move="pbyp" gpu="yes">
    <estimator name="LocalEnergy" hdf5="no"/>
    <parameter name="nonlocalmoves"> yes </parameter>
    <parameter name="warmupsteps">100</parameter>
    <parameter name="blocks">DMCBLOCKS</parameter>
    <parameter name="steps">10</parameter>
    <parameter name="timestep">0.02</parameter>
</qmc>
</simulation>
```

The key parameters to be replaced are

- TITLE: title of a run
- SEED: random number seed. Default SEED=-1 generates random number seeds based on the time of a run
- WFSXML: input XML to define a trial wavefunction
- VMCBLOCKS: number of VMC blocks
- VMCSTEPS: number of VMC steps per block
- WALKERS: number of walkers during a VMC section.
- SAMPLES: number of walker configurations to be stored in memory to perform the next DMC section. This is the target population of the main DMC.
- DMCBLOCKS: number of DMC blocks.

This run will dump configurations at every 100 blocks but it can be turned off by setting `checkpoint="-1"`.

- `gr3x3x1.wfs.xml`: define the trial wavefunction for a small benchmark problem

- gr4x4x1.wfs.xml: define the trial wavefunction for a big benchmark problem
- lda.pwscf.h5: portable binary file containing the single-particle orbitals in the spectral space
- C.BFD.xml: Pseudopotential file.

Q2 Run Script - CPUs This example uses 256 cores and OMP_NUM_THREADS=8. The size of MPI is set according to these variables. The sample size is determined by the number of cores used for a run. 4 walkers per core (see export twalkers=4) is recommended for large problems.

```
#!/bin/bash
#PBS -A CSC091qmcpac
#PBS -N debug
#PBS -j oe
#PBS -l walltime=00:30:00,size=256
##PBS -l feature=gpu
#PBS -l gres=widow3
#PBS -V

export NUM_CORES_PER_SMP=1
export COMPCORES=1

cd $PBS_O_WORKDIR

export HUGETLB_MORECORE=yes
export OMP_NUM_THREADS=8
export MPICH_RANK_REORDER_DISPLAY=1
export COUNTER1=GET_TIME_OF_DAY

export MPICH_PTL_SEND_CREDITS=-1
export MPICH_MAX_SHORT_MSG_SIZE=1024
export MPICH_PTL_UNEX_EVENTS=800000
export MPICH_UNEX_BUFFER_SIZE=16M
export MPI_MSGS_PER_PROC=32768

let NP=$PBS_NNODES/$OMP_NUM_THREADS/$COMPCORES
let NPPNODE=2

export qmcexe=/lustre/widow3/scratch/jnkim/xk6/qmcpack-trunk/bin/qmcapp
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
export WFSXML=gr4x4x1.wfs.xml
export twalkers=4
export VMCBLOCKS=10
let VMCSTEPS=$twalkers*$VMCBLOCKS/10
export DMCBLOCKS=100
export VMCWALKERS=1
let SAMPLES=$NP*$OMP_NUM_THREADS*$VMCBLOCKS*$VMCWALKERS

#####
# create a directory to run the job
# replace KEYWORDS with the run parameters and create input.xml
#####
export title=t.gnu46.gr4x4x1.p${NP}x${OMP_NUM_THREADS}.w${twalkers}.`date +"%m-%d-%y_%H%M"`
mkdir -p $title
cp $WFSXML $title/

cat qmc.xml \
| sed s/TITLE/bench/ \
| sed s/WFSXML/$WFSXML/ \
| sed s/VMCBLOCKS/$VMCBLOCKS/ \
| sed s/VMCSTEPS/$VMCSTEPS/ \
| sed s/WALKERS/$VMCWALKERS/ \
| sed s/SAMPLES/$SAMPLES/ \
| sed s/DMCBLOCKS/$DMCBLOCKS/ \
| sed s/SEED/-1/ \
> ${title}/input.xml
cd ${title}

aprun -n ${NP} -S 1 -d ${OMP_NUM_THREADS} ${qmcexe} input.xml &> qmc.log
```

Q2 Run Script - GPUs This is essentially the same as the script for CPU runs. The only difference is the number of walkers for the VMC section and target walkers per GPU. To obtain good efficiency on XK6 nodes with GPUs, more than 32 walkers per GPU is recommended. This example uses 128 walkers for both VMC and DMC. Note that OMP_NUM_THREADS=1.

```
#!/bin/bash
#PBS -A CSC091qmcpac
#PBS -j oe
#PBS -l feature=gpu
#PBS -l walltime=00:30:00,size=CPUCORES
#PBS -l gres=widow3
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
#PBS -V

module load craype-accel-nvidia20

export NUM_CORES_PER_SMP=1
export OMP_NUM_THREADS=1
let NP=$PBS_NNODES/$OMP_NUM_THREADS/$COMPCORES

cd $PBS_O_WORKDIR
export HUGETLB_MORECORE=yes
export MPICH_RANK_REORDER_DISPLAY=1
export COUNTER1=GET_TIME_OF_DAY

export MPICH_PTL_SEND_CREDITS=-1
export MPICH_MAX_SHORT_MSG_SIZE=1024
export MPICH_PTL_UNEX_EVENTS=800000
export MPICH_UNEX_BUFFER_SIZE=16M
export MPI_MSGS_PER_PROC=32768

export qmcexe=/lustre/widow3/scratch/jnkim/xk6/qmcpack-trunk-cuda/bin/qmcapp

export WFSXML=gr4x4x1.wfs.xml

export twalkers=128
export VMCBLOCKS=10
let VMCSTEPS=$VMCBLOCKS/10
export DMCBLOCKS=100
export VMCWALKERS=$twalkers
let SAMPLES=$NP*$VMCWALKERS

export title=t.cuda.gr4x4x1.p${NP}x${OMP_NUM_THREADS}.w${twalkers}.`date +"%m-%d-%y_%H%M"`
mkdir -p $title
cp $WFSXML $title/

cat qmc.xml \
| sed s/TITLE/bench/ \
| sed s/WFSXML/$WFSXML/ \
| sed s/VMCBLOCKS/$VMCBLOCKS/ \
| sed s/VMCSTEPS/$VMCSTEPS/ \
| sed s/WALKERS/$VMCWALKERS/ \
| sed s/SAMPLES/$SAMPLES/ \
| sed s/DMCBLOCKS/$DMCBLOCKS/ \
```

US DOE SC ASCR FY12 Joule Software Metric: Q2 Status Report

```
| sed s/SEED/-1/ \
> ${title}/input.xml

cd ${title}
aprun -n ${NP} -N 1 -d ${OMP_NUM_THREADS} ${qmcexe} --gpu input.xml &> qmc.log
```